Computationally Efficient Sparse Prior in Regularized Iterative Tomographic Reconstruction

Thibault Notargiacomo

Communauté
UNIVERSITÉ Grenoble Alpes

THÈSE

Pour obtenir le grade de

DOCTEUR DE LA COMMUNAUTÉ UNIVERSITÉ GRENOBLE ALPES

Spécialité : SIGNAL IMAGE PAROLE TELECOMS

Arrêté ministériel : 25 mai 2016

Présentée par

Thibault NOTARGIACOMO

Thèse dirigée par **DOMINIQUE HOUZET** et
codirigée par **Vincent FRISTOT**

préparée au sein du **Laboratoire Grenoble Images Parole Signal Automatique**
dans **l'École Doctorale Electronique, Electrotechnique, Automatique, Traitement du Signal (EEATS)**

Approche parcimonieuse et calcul haute performance pour la tomographie itérative régularisée.

Thèse soutenue publiquement le **14 février 2017**,
devant le jury composé de :

Madame Françoise PEYRIN
Directeur de recherche, INSERM, Rapporteur
Monsieur Frédéric MAGOULES
Professeur, Centrale Supélec, Rapporteur
Monsieur Simon RIT
Chargé de recherche, CNRS, Examinateur
Monsieur Nicolas GAC
Maître de conférences, Université Paris Sud 11, Examinateur
Monsieur Laurent DESBAT
Professeur des universités, Université Grenoble Alpes, Président

UNIVERSITÉ DE GRENOBLE

ÉCOLE DOCTORALE EEATS

Electronique, Electrotechnique, Automatique, Traitement du Signal

T H È S E

pour obtenir le titre de

docteur en sciences

de l'Université Grenoble Alpes

Mention : spécialité Signal, Image, Parole, Télécom (SIPT)

Présentée et soutenue par

Thibault NOTARGIACOMO

Computationally Efficient Sparse Prior in Regularized Iterative Tomographic Reconstruction

Thèse dirigée par Dominique Houzet

préparée au laboratoire de recherche Grenoble Images, Parole, Signal, Automatique GIPSA Lab

soutenue le 14 Févier 2017

Jury :

Rapporteurs :	Françoise Peyrin	-	INSERM, Laboratoire Creatis
	Frédéric Magoulès	-	Centrale-Supelec, Laboratoire MICS
Directeur :	Dominique Houzet	-	Grenoble INP, Laboratoire GIPSA-Lab
Encadrant :	Vincent Fristot	-	Grenoble INP, Laboratoire GIPSA-Lab
	Guillaume Bernard	-	Thales TED Moirans
Président :	Laurent Desbat	-	Université Grenoble Alpes, Laboratoire TIMC-IMAG
Examinateur :	Simon Rit	-	CNRS, Laboratoire Creatis
	Nicolas GAC	-	Paris Sud 11, Laboratoire des signaux et systèmes

Acknowledgments

The author would like to thanks the Gipsa-Lab for providing a motivating work environment, and gathering in the same place so many signal processing and optimization enthusiasts. In particular, we would like to thank Laurent Condat for organizing its optimization workshop, featuring distinguished speakers as well as stimulating teaching material, and hands-on lab.

We also want to thank Thales for providing, in addition to the fundings, an appropriate management, and the freedom to explore the research field of inverse problem and its numerous extensions.

We would like to highlight the fact that working in the Rhône-Alpes geographic area was a rewarding experience, due to its concentration of highly talented people and new technology enterprises, devoted to share knowledge and experience in science and engineering. In this context, I wanted to thank Laurence Viry for managing the Grenoble-calcul expert group in the domain of high performance computing, as well as people from the Lyon-calcul group and Bull. I also wanted to thank Simon Rit and Cyril Mory, from the Creatis laboratory, that guided me in the world of tomography, and for their effort in bringing quality opensource code to the world, through RTK, with a very active support.

My knowledge in signal processing and tomography would have remained poor if I had not met Pr Laurent Desbat, whose teaching skills are remarkable.

Finally, this thesis would have remained a dream if not for the vision, the optimism, and the enthusiasm of Guillaume Bernard, which turned this project into a reality in the fall of 2013, along with Albert Murienne who helped me with technical issues during the thesis, both invested their time and energy to design, build and operate the tomographic platform in Moirans, so that we were able to challenge reconstruction of real datasets.

Contents

List of Figures

[1] *Discrete Fourier Transform*

List of Tables

List of Acronyms

ANR	*Agence Nationale de la Recherche*
ANRT	*Association Nationale de la Recherche et de la Technologie*
CEA	*Commissariat à l'énergie atomique et aux énergies alternatives*
CIFRE	*Conventions Industrielles de Formation par la REcherche*
List	*Laboratoire d'Intégration de Systèmes et des Technologies*
TED	*TED Thales Electron Devices SAS*
RTK	*Open Reconstruction ToolKit*
DFT	*Discrete Fourier Transform*
FFT	*Fast Fourier Transform*
RBF	*Radial Basis Function*
PSWF	*Prolate Spheroidal Wave Functions*
VST	*Variance Stabilizing Transformation*
ADMM	*Alternating Direction Method of Multipliers*
POCS	*Projection Onto Convex Sets*
EM	*Expectation Maximization*
KL	*Kullback-Leibler*
BB	*Barzilai-Borwein*
TV	*Total Variation*
DTCWT	*Dual-Tree Complex Wavelet Transform*
RMF	*Random Markov Field*
CC	*Cartesian Cubic Lattice*
BCC	*Body Centered Cubic Lattice*
FCC	*Face Centered Cubic Lattice*
ECC	*Epipolar Constistency Conditions*
AABB	*Axis Aligned Bounding Box*

DOF	*Degree Of Freedom*
CT	*Computed Tomography*
CBCT	*Cone Beam Computed Tomography*
ART	*Algebraic Reconstruction Technique*
SART	*Simultaneous Algebraic Reconstruction Technique*
SIRT	*Simultaneous Iterative Reconstruction Technique*
SIR	*Statistical Iterative Reconstruction*
FBP	*Filtered Back Projection*
FDK	*Feldkamp, Davis and Kress method*
NDT	*Non Destructive Testing*
PET	*Positron Emission Tomography*
kV	*kilo-Volts*
FPD	*Flat Panel Detector*
DQE	*Detective Quantum Efficiency*
CUDA	*Compute Unified Device Architecture*
DSL	*Domain-Specific Language*
GPGPU	*General-Purpose Computing on Graphics Processing Units*
GPU	*Graphics Processing Unit*
SM	*Streaming Multiprocessor*
SFU	*Special Function Unit*
SIMD	*Single Instruction Multiple Data*
ALU	*Arithmetic Logic Unit*
FPU	*Floating Point Unit*
TBB	*Thread Building Block*

Chapter 1: Introduction

Sommaire

1.1 Motivations and Context

This work was financially suppoted by France CIFRE[1] convention Number 2013/0971 and TED[2] Moirans, and follows a seminal work conducted by Han Wang (CIFRE 220/2008), see [**wang2011methodes**].

The main idea of the project is to exploit the latest advances in the field of applied mathematics and computer sciences in order to study, design and implement algorithms dedicated to 3D cone beam reconstruction from X-Ray flat panel detectors targeting clinically relevant usecases.

Among the most iconic breakthrough that motivated this work, we can cite the birth of the compressed sensing theory in 2006, see [**candes2006robust**] and the first release of the CUDA[3] devlopment kit, a proprietary DSL[4] from NVidia, dedicated to GPGPU[5] in 2007.

Those breakthrough, although not directly related to each other, resulted in numerous subsequent discoveries or new applications in the field of randomized linear algebra, graph

[1] *Conventions Industrielles de Formation par la REcherche*
[2] *TED Thales Electron Devices SAS*
[3] *Compute Unified Device Architecture*
[4] *Domain Specific Language*
[5] *General Purpose Computing on Graphics Processing Units*

theory, large scale optimization, computer vision, machine learning and found applications in many other fast growing fields of the digital industry.

In this study, we will restrict ourselves to the problem of CBCT[6] reconstruction, and we will try to make some links with the aforementioned fields when needed.

1.1.1 Computed tomography

Computed tomography is a technique that aims to provide a measure of a given property of the interior of a physical object, given a set of exterior projection measurement.

Although multiple types of tomography exists, based on positron emission, fluoresence, electron beam, seismic imaging, we will restrict our attention to the most widely used modality which is the X-Ray transmission tomography for density reconstruction. CT is a mature technology, and examples of the use of such techniques in the everyday life arise in industry, for NDT[7] but also fo security check in aiports, or dental and angiograpic imaging in hospitals.

We can summarize this modality as a two step method:

- Acquisition: the system acquire a set of X-Ray transmission images along a trajectory, either dictated by the movment of the X-Ray source, the detector, the object or a specific combination of the three elements.
- Reconstruction: For every point of the space, that can lie over a predefined grid, or an adapted representation system, an attenuation coefficient is derived such that those data must be consistent with the acquisition data, the physical and geometrical property of the acquisition model, and eventually some apriori.

In this thesis, we will mostly restrict our attention to the second point.

1.1.2 Clinical interest

This PhD project was carried out in the framework of the ANR[8] Voxelo, see [**ANRVoxelo**], in collaboration with the CEA[9] List[10], which aimed at providing CBCT reconstruction algorithm, exploiting few views for the diagnosis of osteoarthritis.

Cone beam tomography, when performed using C-Arm or dedicated extremity imaging systems [**carrino2013dedicated**], is potentially less expensive than helical CT, and may allow a generalization of 3D imaging to annual screening of degenerative bone disease like

[6] *Cone Beam Computed Tomography*
[7] *Non Destructive Testing*
[8] *Agence Nationale de la Recherche*
[9] *Commissariat à l'énergie atomique et aux énergies alternatives*
[10] *Laboratoire d'Intégration de Systèmes et des Technologies*

osteoporosis or osteoarthritis. Frequent assessment of anatomical metrics over bones and joints motivates the design of reconstruction algorithms capable of accurate retrieval of bone microstructure from low dose acquisitions.

1.2 Previous work

1.2.1 Analytical approach

The mathematics of analytical tomography are known for a long time, the first work on this topic dates back to 1917, when Johann Radon introduced the 2D Radon transfom, along with its inversion formula. In this work, the solution for the inversion of the Radon transform, and subsequent analysis heavily relies on the pojection-slice theorem. It is worth noting that the discretization of the Radon transform in 2D for parallel and fan beam geometry gave bith to the practical FBP[11] algorithm.

Extensions of this work to higher dimensional spaces led to interesting results in the field of integral geometry, in particular the study of the reconstruction of 3D functions led to multiple devlopment related to our problem.

Among these, we can mention the work of Tuy in [**tuy1983inversion**] that established an inversion formula related to a model of acquisition trajectory in 3D, in addition to a set of condition over this trajectory to ensure proper reconstruction. Grangeat in [**grangeat1991mathematical**] also exposed an exact reconstruction formula based on the derivative of the Radon transform. But one of the most widely used method, compatible with an easy and fast implementation was the FDK[12], which is an appoximate formula, exposed in [**feldkamp1984practical**].

1.2.2 Algebraic approach

1.2.2.1 Analytical approach limitations

Although analytical methods are generally robust, rely on very few filtering parameters, and support extremely fast implementations, able to reconstruct volumes in the matter of seconds, they suffer some limitations.

The first limitation is related to the management of missing data: the analytical method does not provide a clear framework to handle incomplete dataset: in the case where there are some invalid pixels in the detector, there does not seem to be a simple image filter, or weighting function allowing to handle such inconsistency in the acquisition model.

[11] *Filtered Back Projection*
[12] *Feldkamp, Davis and Kress method*

One annoying limitation is related to the modelization of the X-Ray transmission image formation process: they are generally unable to account for informations known a piori, like the size of the X-Ray source focal spot, its emission spectrum, or the spectral sensitivity of the detector, the statistical property of the X-Ray flux, or the detector noise.

Another important features that analytical approaches lack, is the ability to incorporate a-priori infomations about the reconstructed volume, often expressed as mathematical property, like regularity, spatial or spectral support, or compressibility.

All the previous features can be incorporated when the tomographic reconstruction problem is recasted as a linear inverse problem, whose resolution rely on tools arising from the field of linear algebra and more generally from the world of convex optimization.

1.2.2.2 First use of algebraic methods in tomography

The use of algebraic method in tomographic reconstruction is relatively new regarding to the devlopment of the algorithm for solving linear problems. This is in part, due to the fact that the discretization of the projection operator, and its transpose, the backprojection operator, cannot be easily written down in their matrix form. The size of such matrices could often exceed 10^{12} to 10^{16} in 3D, even if their sparsity lead to record only 10^9 to 10^{10} non zero terms, such problem remained numerically intractable for a long time without the use of dedicated methods.

The fact that projection and backprojection linear operators can be computed on the fly, however, allowed researchers to experiments with some algorithms known for their fast convergence, since 1970.

One of the first algebraic resolution method that have been applied to the problem of tomographic reconstruction was ART[13], see [**gordon1970algebraic**] latter followed by SIRT[14] [**gilbert1972iterative**] and, a few years later, the famous SART[15], see [**andersen1984simultaneous**].

The ART method was actually an instance of the Kaczmarz method applied to tomography, which itself is an instance of a more general method called POCS[16], SART and SIRT can be view as relaxed versions of the Kaczmarz.

As every pixel of the projection data defines a linear equality constraint, the solution of the problem should be seen as a point in a high dimensional space, lying on the interesection of all hyperplans defined by the equality constraints. As hyperplans are simple convex sets, ART and its variant can be seen as geometric method projecting a current solution over successive hyperplans, or moving in the direction of the barycenter of multiple hyperplans projections.

[13] *Algebraic Reconstruction Technique*
[14] *Simultaneous Iterative Reconstruction Technique*
[15] *Simultaneous Algebraic Reconstruction Technique*
[16] *Projection Onto Convex Sets*

1.2.2.3 Statistical methods

Although SART, the subset variant of ART met an important success in the tomography community, providing a relatively fast convergence, hence allowing to deal more elegantly with missing or inconsistent data, it was not able to handle physical model of X-Ray images formation properly as is.

This gave rise to a new paradigm in the tomography community, known as SIR[17] or someteimes as model-based reconstruction although the later is more general. One of the first statistical reconstruction method was designed in the framework of PET[18] imaging, and was simply an instance of the well known EM[19] algorithm, see [**lange1984reconstruction**]. It aimed at maximizing the likelihood of the projection data, or equivalent minimizing their KL[20]-divergence, using the a-priori Poisson statistics of radionuclide decay over a set of unobserved data.

Using this paradigm, many other likelihood formulation were derived, and solved using either EM or gradient ascent like algorithms, we can cite for instance the use of noise statistics for X-Ray transmission tomography [**erdogan1999ordered**], and even the derivations of a statistical flavour of reconstructed volume regularity based on a Gibbs prior over a RMF[21], see [**bouman1993generalized**].

1.2.3 Regularization in iterative image reconstruction

Statistical apriori regarding projection noise, X-Rays spectrum, and other physical property has been refined along the years, but a large number of studies and innovations arose from the more general class of regularized iterative reconstruction. Regularized tomographic reconstruction method are based on the algebraic formulation, they usually consist in optimizing a composite objective function, preferably convex, which embeds informations about projection data fidelity, and a metric measuring discrepancy with an apriori model of the data to be reconstructed.

The litterature about tomographic reconstruction regularization is vast, and a lot of models have been proposed, for instance we can cite the total variation model [**rudin1992nonlinear**], the heuristic of patch based redundancy [**xu2009performance**], wavelet sparsity [**yazdanpanah2016sparse**], dictionary based sparsity [**zhang2016low**], and more recently non local total variation [**kim2016non**], adaptive graph based total variation [**mahmood2016compressed**], and low rank approximation [**ongie2016giraf**].

It is interesting to see that sparsity priors attracted a lot of attention from the tomographic reconstruction community, this success is probably a related to the discovery of the uncertainty

[17] *Statistical Iterative Reconstruction*
[18] *Positron Emission Tomography*
[19] *Expectation Maximization*
[20] *Kullback Leibler*
[21] *Random Markov Field*

principle exposed by Donoho in [**donoho2001uncertainty**], and later exploited by Candes in its Compressive sensing theory [**candes2006robust**].

1.3 Contributions

In this work, we propose to study multiple approach for leveraging GPU computing and sparsity promoting priors in CBCT.

Signal processing considerations regarding the acquisition system modelization, and the spatial discretization model will be adressed in chapter 2. Challenges related to tomographic operator modelization, and implementation, along with geometric framework will be introduced in chapter 3.

In the chapter 2, we propose to study how the X-Ray acquisition process can be modeled, in particular we will focus on how data can be discretized, and analyze various solutions from a signal processing point of view.

In chapter 3, we will study how tomographic operators can be modeled and implemented using high performance hardware, and propose a simple framework allowing to project smooth radially symmetric volume elements called blobs.

The next chapter 4 is dedicated to the study of the interaction of tomographic models with various first order optimization methods to solve the algebraic formulation of the CBCT problem.

In the last chapter 5, we will focus on the use of sparse priors for CBCT reconstruction, and develop a new reconstruction method based on 3D complex directional wavelets.

Chapter 2: Problem definition, and object of the study

Sommaire

2.1 X-ray image formation modelization

2.1.1 Basics of X-ray generator

Most of the medical CT system use a X-Ray generator containing an X-Ray tube. Those devices are actually vacuum tube containing two electrod, connected to a high voltage power source, usually between 50 kV[1] and 150 kV that generates a large electric potential difference.

The cathode often contains a coil, heated to high temperatures ($700 - 800°$) by a high intensity electric current such that a flux of electrons is emitted due to thermionic effect. The electrons expelled from the cathode are then accelerated to relativistic speeds towards the

[1] *kilo Volts*

anode due to the high electric potential difference. The point where the high energy electrons beam collides with the anode is the place where the emission of the X-Ray take place, and is called the focal spot.

The X-Ray spectrum depends on the anode material, and the electric potential difference that accelerated the electrons, usually, one would seek for the smallest possible focal spot, as in a convergent optical system, such that the image will not suffer from blur at the detector spot, we illustrated this issue on figure 2.1.

The derivation of accurate X-Ray projection model, accounting for the focal spot blur is an active topic of research, see for instance [**tilley:16:msv**].

It is also worth noting that, for a given electron flux, the thinner the focal spot is, the higher the temperature of the anode material is, at this point. And high temparature often result in a broader X-Ray spectrum, which is something we would like to avoid, as we will see in the next section.

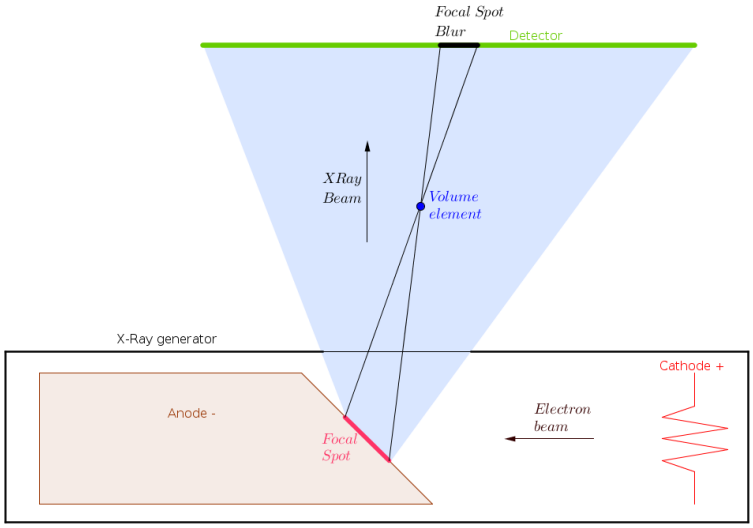


Figure 2.1: Simplified schematic of a X-Ray generator

2.1.2 Deriving photon count from attenuation

In order to model the image formation in X-Ray transmission imaging, it is important to understand at a very basic level, how the X-Ray beam propagates and interact with matter before hitting the detector, so that we will be able to recover information about the target object.

We will first assume that, given a time slice of duration T, our source S is able to generate $n_S i(E)$ photons, of energy E, whose trajectory, assumed straight, will cross the surface of the i^{th} detector bin .

Now, during the experiment, those photons will travel through a 3D space, whose linear

attenuation coefficient at position x, will be equal to $\mu(x, E)$. Following the model presented in [**lange1984reconstruction**], we can imagine that our $n_S i(E)$ independant photons experience a Bernouilli process, with two posible outcomes:

- The photon travels through the medium, along a straight line L_i without interacting, and reach the detector with a probability $p_i(E) = e^{-\int_{L_i} \mu(x,E)dx}$
- The photon interacted with the medium, somewhere along its path, and did not reached the detector with a probability $1 - p_i(E)$

The formula used for $p_i(E)$ was simply derived from the well known Lambert's law.

Let $Y_i(E)$ be the random variable describing the number of photons hitting the i^{th} detector bin, it is given by a sum of Bernouilli's process :

$$Y_i(E) = \sum_{j=0}^{n_S i(E)-1} Y_{ij}(E) \tag{2.1}$$

with $Y_{ij}(E)$ the binary random variable desribing the fact that the j^{th} photon reached the i^{th} detector bin, and

$$P(Y_{ij}(E) = 1) = p_i(E) \tag{2.2}$$
$$P(Y_{ij}(E) = 0) = 1 - p_i(E) \tag{2.3}$$
$$\tag{2.4}$$

Le Cam showed that the probability density functions of sums of Bernouilli's variable such that the one described in equation 2.1 converges toward the following Poisson law of parameter $\lambda > 0$:

$$P(Y_i(E) = k) = \frac{\lambda^k}{k!} e^{-\lambda} \tag{2.5}$$

Where $\lambda = \mathbb{E}[Y_i(E)] = n_S i(E) p_i(E)$

2.1.3 Noise modeling in image formation

The statistical model seen in section 2.1.2, can be further refined, if we take into account a random noise model for the detector. The most common noise model used in digital imaging with flat panel detectors, can be derived from the physic of semiconductor and often relates to an additive centered gaussian noise, with standard deviation σ.

In this case, we have a new random variable $Z_i(E)$ which is equivalent to the number of photons actually detected by detector bin i. Its statistical model actually depends on the value of $Y_i(E)$, which for the record, is the number of photons hitting the detector surface :

$$P(Z_i(E) = k|Y_i(E) = l) = e^{-\frac{(l-k)^2}{2\sigma^2}} \tag{2.6}$$

the probability that the i^{th} detector bin actually reports a photon count equal to k can be expressed using the following marginal distribution:

$$P(Z_i(E) = k) = \sum_{l=0}^{+\infty} P(Z_i(E) = k|Y_i(E) = l)P(Y_i(E) = l) \tag{2.7}$$

$$= \sum_{l=0}^{+\infty} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(l-k)^2}{2\sigma^2}} \frac{\lambda^l}{l!} e^{-\lambda} \tag{2.8}$$

Of course, in this case where gaussian and poissonian statistics are mixed, it may be hard to retrieve the noise parameters. In particular, applying VST[2], in order to be able to use reconstruction method suited for homoscedastic data may be hazardous. Some studies showed how to assess FPD[3] noise statistics in such challenging cases, see [**hsieh2015compound**].

It can be noticed that more accurate statistical model can be derived for non photon counting detector using apriori knowledge about scintillator physical behaviour, and quantization error of the digital to analog converters. Unfortunately infering all these model parameters, need specific experiments that are beyond the scope of this work.

2.1.4 A simple linear model

In this thesis, we will mostly restrict ourselves to algebraic formulation without statistical priors, it means that we will not consider the Poisson statistics of the input data, nor we will consider the gaussian noise degrading the image, as seen in the model presented in section 2.1.3.

Instead, our study will only consider that each event realization was the most probable case for the random variables $Y_i(E)$ and $Z_i(E)$. Although we drop here some informations about the physical system, our calculation will be simplified, and the methodological bias will not be too important, as, for both statistical model, the most probable event coincide with the expectation of the random variable.

Our ultra simplified estimator for the number of photons detected by the sensor i : $y_i(E)$ then reads

[2] *Variance Stabilizing Transformation*
[3] *Flat Panel Detector*

$$y_i(E) = n_S i(E) e^{-\int_{L_i} \mu(x,E)dx} \tag{2.9}$$

Where $n_S i(E)$ will be estimated from an acquisition with the exact same system, same geometry, but without object, having X-Rays going through a medium with negligible attenuation, air for instance.

It should be noticed, that in real cases, for detectors that are not photon counting capable, we will assume that the sensors are calibrated such that their output value is equal to the number of detected photons up to a multiplicative ratio, accounting for the gain G_i and the quantum efficiency $Q_i(E)$ that are assumed to be known priors.

In practice, the more the DQE[4] of the sensor is close to an ideal detector, the more our simplified estimator will be suited for accurate reconstruction.

We can then derive a linear relationship between the attenuation of the medium, and the signal of the detector:

$$n_S i(E) e^{-\int_{L_i} \mu(x,E)dx} = y_i(E) \tag{2.10}$$

$$e^{-\int_{L_i} \mu(x,E)dx} = \frac{y_i(E)}{n_S i(E)} \tag{2.11}$$

$$\int_{L_i} \mu(x,E)dx = -log\left(\frac{y_i(E) G_i Q_i(E)}{n_S i(E) G_i Q_i(E)}\right) \tag{2.12}$$

$$\int_{L_i} \mu(x,E)dx = -log\left(\frac{p_i(E)}{p0_i(E)}\right) \tag{2.13}$$

Where $p_i(E)$ and $p0_i(E)$ are respectively the digital detector value for pixel i while imaging object and while imaging air.

In this work, we did not focused on spectral reconstruction of the target, which is a specific topic that requires aprioi knowledges about the spectrum of the X-Ray generator, and the quantum efficiency of the detector, we will then even simplify our model towards a monochromatic imaging model:

$$\int_{L_i} \mu(x)dx = -log\left(\frac{p_i}{p0_i}\right) \tag{2.14}$$

[4] *Detective Quantum Efficiency*

2.2 Signal processing tools for problem discretization

2.2.1 Detector discretization

Most of the digital detectors used in X-Ray imaging have a rectangular shape, and, their sensitive area is made of a set of square picture elements, called pixels. If we neglect the quantization noise, the digital imaging process can be modeled as a two step sampling method in the signal processing framework:

- The continuous image projection function $p(x)$ over $\mathbb{R}^2$, is convolved with the function $\chi_{pix}(x)$, which is the pixel indicator function of pixel $(0,0)$ centered over the coordinates $\begin{pmatrix}0\\0\end{pmatrix}$

- The result is multiplied in direct space with the detector grid sampling operator $S_{M_{detector}}(x) = \sum_{k,l} \delta(M_{detector} \begin{pmatrix}k\\l\end{pmatrix} - x)$. With δ the Dirac distribution, and k, l the number of pixels respecticely in x and y direction.

Where $M_{detector}$ is a diagonal matrix that account for the pixel size. Here, the image sampling model may be subject to aliasing, because the Fourier transform of the pixel indicator is a tensor product of cardinal sine, which is not a good lowpass filter that cancels out frequencies outside of the Shanon limit. Unfortunately, this issue is inherent to the detector, and cannot be corrected by the use of another mathematical model during the reconstruction process.

In the following pages, we will only be talking about the volume discretization, because there is no real physical constraint behind the model choice, we will be able to study the various possible solutions.

2.2.2 Fourier theory in 3 dimensions

Although radon transform inversion has been first derived in the framework of continuous functions, we should keep in mind that we would like to be able to model our volume into a set of small volume elements called voxels, lying over the nodes of a regular grid.

In order to understand the importance of the sampling grid in 3D volume discretization, one has to first look at how Fourier series and Shannon theorem reads in higher dimensions.

2.2.2.1 Multidimensional Fourier series

2.2.2.2 Simple orthogonal periodicity

In this section, we will consider a function x in the space of periodic and square integrable functions of n variables:

$$\mathbb{L}^2_P(T_1, T_2, \ldots, T_n) \tag{2.15}$$

such that x, is defined as:

$$x : \underset{(t_1,t_2,\ldots,t_n)}{\mathbb{R}^n} \underset{\rightarrow}{\rightarrow} \underset{x(t_1,t_2,\ldots,t_n)}{\mathbb{R}} \tag{2.16}$$

where $T_1, T_2, \ldots, T_n \in \mathbb{R}^{n*}_+$ are the period along each variables of coordinate of $\mathbb{R}^n$, for which the following periodicity property holds:

$$x\left(\begin{pmatrix} t_1 \\ t_2 \\ \vdots \\ t_n \end{pmatrix} + \begin{pmatrix} T_1 & 0 & \ldots & 0 \\ 0 & T_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & T_n \end{pmatrix} \begin{pmatrix} k_1 \\ k_2 \\ \vdots \\ k_n \end{pmatrix}\right) = x \begin{pmatrix} t_1 \\ t_2 \\ \vdots \\ t_n \end{pmatrix} \tag{2.17}$$

or, using another notation:

$$x(\vec{t} + D_{T_1,T_2,\ldots,T_n}\vec{k}) = x(\vec{t}) \text{ with } \vec{k} \in \mathbb{Z}^n \tag{2.18}$$

$D_{T_1,T_2,\ldots,T_n}$ being the matrix of periodicity that describe the behaviour of the function. In order to use Fourier theory tools in the common functional spaces, we impose that functions of $\mathbb{L}^2_P(T_1, T_2, \ldots, T_n)$ are square integrable over one "'hyper period "':

$$\int_0^{T_1} \int_0^{T_2} \cdots \int_0^{T_n} |x(\vec{t})|^2 \, \mathrm{d}\vec{t} \tag{2.19}$$

We also define $E_{k_1,k_2,\ldots,k_n} \in \mathbb{L}^2_P(T_1, T_2, \ldots, T_n)$ as:

$$E_{k_1,k_2,\ldots,k_n}(t_1, t_2, \ldots, t_n) = E_{\vec{k}}(\vec{t}) \tag{2.20}$$

$$= e^{2i\pi\left(\frac{k_1 t_1}{T_1} + \frac{k_2 t_2}{T_2} + \cdots + \frac{k_n t_n}{T_n}\right)} \tag{2.21}$$

$$= e^{2i\pi \vec{k}^\intercal D^{-1}_{T_1,T_2,\ldots,T_n} \vec{t}} \tag{2.22}$$

We can now prove easily that $E_{k_1,k_2,\ldots,k_n}$ can be used to form an orthonormal basis of $\mathbb{L}^2_P(T_1,T_2,\ldots,T_n)$ using the classical dot product definition in complex functional spaces (hermitian form):

$$\langle E_{\vec{k}}, E_{\vec{l}}\rangle_{\mathbb{L}^2_P(T_1,T_2,\ldots,T_n)} = \int_{0,0,\ldots,0}^{T_1,T_2,\ldots,T_n} E_{\vec{k}}(\vec{t})\,\overline{E_{\vec{l}}(\vec{t})}\,d\vec{t} \tag{2.23}$$

$$= \int_0^{T_1}\int_0^{T_2}\cdots\int_0^{T_n} e^{-2i\pi t_1\frac{l_1-k_1}{T_1}} e^{-2i\pi t_2\frac{l_2-k_2}{T_2}}\ldots e^{-2i\pi t_n\frac{l_n-k_n}{T_n}}\,\mathrm{d}t_1 dt_2\ldots dt_n \tag{2.24}$$

$$= \int_0^{T_1} e^{-2i\pi t_1\frac{l_1-k_1}{T_1}}\,\mathrm{d}t_1 \tag{2.25}$$

$$\times \int_0^{T_2} e^{-2i\pi t_2\frac{l_2-k_2}{T_2}}\,\mathrm{d}t_2 \tag{2.26}$$

$$\vdots \tag{2.27}$$

$$\times \underbrace{\int_0^{T_n} e^{-2i\pi t_n\frac{l_n-k_n}{T_n}}\,\mathrm{d}t_n}_{=\begin{cases} T_n & \text{if } k_n = l_n \\ 0 & \text{otherwise}\end{cases}} \tag{2.28}$$

So we can see that the family $\frac{1}{\sqrt{T_1T_2\ldots T_n}}E_{k_1,k_2,\ldots,k_n}$ forms an orthonormal basis of $\mathbb{L}^2_P(T_1,T_2,\ldots,T_n)$ Now we can derive the multidimensional expression of the Fourier serie of x:

$$x(t_1,t_2,\ldots,t_n) = \sum_{(k_{1i},k_{2j},\ldots,k_{nl})\in\mathbb{Z}^n} c_{k_{1i},k_{2j},\ldots,k_{nl}}(x)E_{k_{1i},k_{2j},\ldots,k_{nl}}(t_1,t_2,\ldots,t_n) \tag{2.29}$$

with

$$c_{k_1,k_2,\ldots,k_n}(x) = \frac{1}{T_1T_2\ldots T_n}\langle x, E_{k_1,k_2,\ldots,k_n}\rangle_{\mathbb{L}^2_P(T_1,T_2,\ldots,T_n)} \tag{2.30}$$

$$= \frac{1}{T_1T_2\ldots T_n}\int_0^{T_1}\int_0^{T_2}\cdots\int_0^{T_n} x(t_1,t_2,\ldots,t_n)\overline{E_{k_1,k_2,\ldots,k_n}}\,\mathrm{d}t_1 dt_2\ldots dt_n \tag{2.31}$$

$$= \frac{1}{T_1T_2\ldots T_n}\int_0^{T_1}\int_0^{T_2}\cdots\int_0^{T_n} x(t_1,t_2,\ldots,t_n)e^{-2i\pi\left(\frac{k_1t_1}{T_1}+\frac{k_2t_2}{T_2}+\cdots+\frac{k_nt_n}{T_n}\right)}\,\mathrm{d}t_1 dt_2\ldots dt_n \tag{2.32}$$

2.2.2.3 Non orthogonal periodicity

Let's now define a more subtle approach of periodicity, for a function x:

$$x : \underset{(t_1,t_2,\ldots,t_n)}{\mathbb{R}^n} \underset{\rightarrow}{\rightarrow} \underset{x(t_1,t_2,\ldots,t_n)}{\mathbb{R}} \tag{2.33}$$

such that

$$x\left(\vec{t} + \sum_{i=1}^{n} k_i \vec{\omega_i}\right) = x(\vec{t}) \tag{2.34}$$

with $W = (\vec{\omega_1}, \vec{\omega_2}, \ldots, \vec{\omega_n})$ a rank-n matrix, hence non singular and forming a basis of $\mathbb{R}^n$:

$$W = \left(\begin{pmatrix}\vec{\omega_1}\end{pmatrix}\begin{pmatrix}\vec{\omega_2}\end{pmatrix}\begin{pmatrix}\vdots\end{pmatrix}\begin{pmatrix}\vec{\omega_n}\end{pmatrix}\right) \tag{2.35}$$

It is interesting to notice that each column vectors from the matrix W corresponds to one of the primitive translation vectors of a lattice. Using a linear combination of these vectors, we can define an hyper-parallelogram $\mathbb{P}_W$ called primitive cell in the framework of lattices, see figure 2.2 for an illustration in dimension 2.

$$\mathbb{P}_W = \vec{v} \in \mathbb{R}^n, \vec{v} = W\vec{u} \text{ with } \vec{u} \in [0,1[^n \tag{2.36}$$

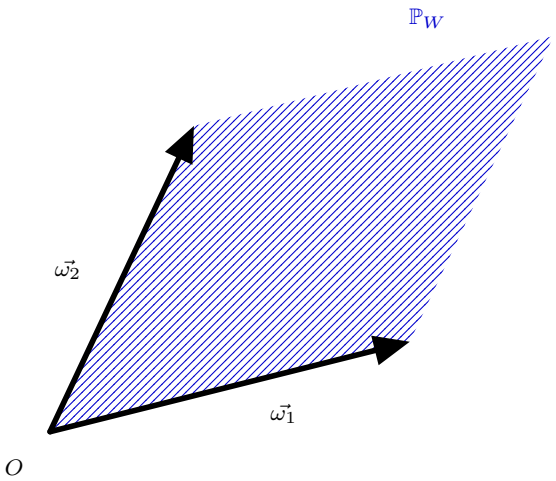


Figure 2.2: Exemple of a primitive cell defined by 2 lattice vectors in 2D

This expression leads to a more general definition of periodicity:
We can say that x is W-periodic if $\forall \vec{k} \in \mathbb{Z}^n$ and $\forall \vec{t} \in \mathbb{R}^n$ we have

$$x(\vec{t} + W\vec{k}) = x(\vec{t}) \tag{2.37}$$

Then the space of square integrable functions over a non orthogonal period is defined as:

$$\mathbb{L}^2_P(W) = x \text{ W-periodic such that } \int_{\mathbb{P}_W} |x(\vec{t})|^2 \, \mathrm{d}\vec{t} < +\infty \tag{2.38}$$

Using a simple vector variable change in the integral over the hyperparallelogram $\mathbb{P}_W$, the inner products for $\mathbb{L}^2_P(W)$ reads:

$$\begin{aligned}
\langle x, y \rangle_{\mathbb{L}^2_P(W)} &= \int_{\mathbb{P}_W} x(\vec{t})\overline{y(\vec{t})}\mathrm{d}\vec{t} && (2.39)\\
&= \int_{\vec{u} \in [0,1[^n} x(W\vec{u})\overline{y(W\vec{u})} \, |det(W)| \, \mathrm{d}\vec{u} && (2.40)\\
&= \int_0^1 \int_0^1 \cdots \int_0^1 x(W\vec{u})\overline{y(W\vec{u})} \, |det(W)| \, du_1 du_2 \ldots du_n && (2.41)
\end{aligned}$$

We can also define a new expression for the trigonometric polynomial used in the Fourier series:

$$\begin{aligned}
E_{k_1,k_2,\ldots,k_n}(t_1, t_2, \ldots, t_n) &= E_{\vec{k}}(\vec{t}) && (2.42)\\
&= e^{2i\pi(\vec{k}^\intercal W^{-1}\vec{t})} \qquad = e^{2i\pi(W^{-t}\vec{k})^\intercal \vec{t}} && (2.43)
\end{aligned}$$

The change of variable $\vec{t} \to W\vec{u}$ used to define the inner product of $\mathbb{L}^2_P(W)$ in 2.2.2.3, can be used to find a nice separable expression of $\langle E_{\vec{k}}, E_{\vec{l}} \rangle_{\mathbb{L}^2_P(W)}$:

$$\langle E_{\vec{k}}, E_{\vec{l}} \rangle_{\mathbb{L}^2_P(W)} = \int_{\mathbb{P}_W} E_{\vec{k}}(\vec{t})\ \overline{E_{\vec{l}}(\vec{t})}\ d\vec{t} \tag{2.44}$$

$$= \int_{\vec{u}\in[0,1[^n} e^{2i\pi(\vec{k}^\intercal W^{-1} W\vec{u})} e^{-2i\pi(\vec{l}^\intercal W^{-1} W\vec{u})}\ |det(W)|\ \mathrm{d}\vec{u} \tag{2.45}$$

$$= |det(W)| \int_{\vec{u}\in[0,1[^n} e^{-2i\pi\left(\vec{l}-\vec{k}\right)^\intercal \vec{u}}\ \mathrm{d}\vec{u} \tag{2.46}$$

$$= |det(W)| \int_0^1 \int_0^1 \cdots \int_0^1 e^{-2i\pi u_1(l_1-k_1)} e^{-2i\pi u_2(l_2-k_2)} \ldots e^{-2i\pi u_n(l_n-k_n)}\ \mathrm{d}u_1 du_2 \ldots du_n \tag{2.47}$$

$$= |det(W)| \int_0^1 e^{-2i\pi u_1(l_1-k_1)}\ \mathrm{d}t_1 \tag{2.48}$$

$$\times \int_0^1 e^{-2i\pi u_2(l_2-k_2)}\ \mathrm{d}t_2 \tag{2.49}$$

$$\vdots \tag{2.50}$$

$$\times \underbrace{\int_0^1 e^{-2i\pi u_n(l_n-k_n)}\ \mathrm{d}t_n}_{=\begin{cases}1 & \text{if } k_n = l_n\\ 0 & \text{otherwise}\end{cases}} \tag{2.51}$$

$$\tag{2.52}$$

We can conclude that the family $\frac{1}{\sqrt{|det(W)|}} E_{\vec{k}}$ forms an orthonormal basis of $\mathbb{L}^2_P(W)$ From there, we can see that the multidimensional expression of the Fourier serie of x exposed in equation 2.2.2.2, is still valid, assuming that the Fourier coefficients now reads:

$$c_{\vec{k}}(x) = \frac{1}{|\det(W)|} \langle x, E_{\vec{k}} \rangle_{\mathbb{L}^2_P(W)} \tag{2.53}$$

2.2.3 Sampling and Nyquist-Shannon frequency

2.2.3.1 From discrete to continuous

For this part, we will consider a specific bandlimited, square integrable, function space called PW for Paley-Wiener:

$$PW_{F_0} = x \in L^2(\mathbb{R}) \quad \text{such that} \quad \hat{x}(f) = 0 \forall f \notin [-F_0, F_0] \tag{2.54}$$

$\hat{x}(f)$ being the Fourier transform of x.

As PW_{F_0} represent a set of continuous function over $\mathbb{R}$, we will be able to define a sampling operator E in conjunction with a sampling frequency η such as:

$$E(x)_n = x\left(\frac{n}{\eta}\right), n \in \mathbb{Z}, x \in PW_{F_0} \tag{2.55}$$

The following part will help us to deduce interesting properties on the sampling frequency: let's define

$$\Gamma(f) = \sum_{k=-\infty}^{+\infty} \hat{x}(f - k\eta) \tag{2.56}$$

Γ is the η-periodic version of $\hat{x}(f)$, ie

$$\Gamma(f) = \hat{x}(f) * \sum_{k=-\infty}^{+\infty} \delta(f - k\eta) \tag{2.57}$$

$*$ denoting the convolution operator and δ the dirac distribution. It is interesting to notice that this convolution with a Dirac comb in Fourier space is equivalent to the sampling operator in the initial space, ie its inverse Fourier transform γ is given by:

$$\gamma(t) = \frac{1}{\eta} x(t) \sum_{k=-\infty}^{+\infty} \delta\left(t - k\frac{1}{\eta}\right) \tag{2.58}$$

whose equivalent definition is given by:

$$\gamma(t) = \begin{cases} \frac{1}{\eta} E(x)_n & \text{if } t = \frac{n}{\eta} \\ 0 & \text{elsewhere} \end{cases} \tag{2.59}$$

As Γ is η periodic, we can express it as a Fourier serie:

$$\Gamma(f) = \sum_{n=-\infty}^{+\infty} c_n(\Gamma) e^{2\pi j n \frac{f}{\eta}} \tag{2.60}$$

$$= \sum_{n=-\infty}^{+\infty} c_{-n}(\Gamma) e^{-2\pi j n \frac{f}{\eta}} \tag{2.61}$$

Its Fourier coefficients are given by

$$c_{-n}(\Gamma) = \frac{1}{\eta}\int_{-\frac{\eta}{2}}^{\frac{\eta}{2}} \Gamma(f) e^{2\pi j n \frac{f}{\eta}}\,\mathrm{d}f \tag{2.62}$$

We can see that, if $\frac{\eta}{2} < F_0$, $\Gamma(f)$ on $[-\frac{\eta}{2}, \frac{\eta}{2}]$ will be polluted by its adjacent copies by convolution, ie $\hat{x}(f) * \delta(f-\eta)$ or $\hat{x}(f) * \delta(f+\eta)$ This problem is called aliasing, and it does not allow for the following developments.

In the case where, $\forall f \in [-\frac{\eta}{2}, \frac{\eta}{2}], \Gamma(f) = \hat{x}(f)$, ie in the case where $\frac{\eta}{2} \geq F_0$, we have :

$$c_{-n}(\Gamma) = \frac{1}{\eta}\int_{-\frac{\eta}{2}}^{\frac{\eta}{2}} \hat{x}(f) e^{2\pi j n \frac{f}{\eta}}\,\mathrm{d}f \tag{2.63}$$

We recognize here the inverse Fourier transform of $\hat{x}$ calculated in $t = \frac{n}{\eta}$ ie:

$$c_{-n}(\Gamma) = \frac{1}{\eta} x\left(\frac{n}{\eta}\right) \tag{2.64}$$

Reinjecting this expression into the Fourier serie gives:

$$\Gamma(f) = \sum_{n=-\infty}^{+\infty} \frac{1}{\eta} x\left(\frac{n}{\eta}\right) e^{2\pi j n \frac{f}{\eta}} \tag{2.65}$$

We established a link between a discrete set of samples $E(x)_n, n \in \mathbb{Z}$ and a periodized version of $\hat{x}(f), f \in \mathbb{R}$ expressed with its Fourier coefficients $c_n, n \in \mathbb{Z}$ under the condition that $\frac{\eta}{2} \geq F_0$.

Now let's see how $x(t)$ can be recovered from $\hat{x}(f)$:

$$\begin{aligned}
x(t) &= \int_{-\infty}^{+\infty} \hat{x}(f) e^{2\pi j f t}\,\mathrm{d}f && (2.66)\\
&= \int_{-\frac{\eta}{2}}^{\frac{\eta}{2}} \Gamma(f) e^{2\pi j f t}\,\mathrm{d}f && (2.67)\\
&= \frac{1}{\eta}\int_{-\frac{\eta}{2}}^{\frac{\eta}{2}} \left(\sum_{n=-\infty}^{+\infty} x\left(\frac{n}{\eta}\right) e^{-2\pi j n \frac{f}{\eta}}\right) e^{2\pi j f t}\,\mathrm{d}f && (2.68)\\
&= \frac{1}{\eta}\sum_{n=-\infty}^{+\infty} x\left(\frac{n}{\eta}\right) \int_{-\frac{\eta}{2}}^{\frac{\eta}{2}} e^{2\pi j f (t-\frac{n}{\eta})} df && (2.69)
\end{aligned}$$

The integral part of the previous expression is equivalent to the inverse Fourier transform of a rectangular function, with a scale of $\frac{\eta}{2}$, whose expression can be derived as follows:

$$x(t) = \frac{1}{\eta} \sum_{n=-\infty}^{+\infty} x\left(\frac{n}{\eta}\right) sinc_{\pi}\left(\eta\left(t - \frac{n}{\eta}\right)\right) \tag{2.70}$$

with $sinc_{\pi}(x) = \frac{sin(\pi x)}{\pi x}$. To summarize what have been said, if $\frac{\eta}{2} \geq F_0$, we can reconstruct perfectly both $x(t)$ and $\hat{x}(f)$ with:

$$x(t) = \sum_{n=-\infty}^{+\infty} x\left(\frac{n}{\eta}\right) sinc_{\pi\eta}\left(t - \frac{n}{\eta}\right) \tag{2.71}$$

$$\hat{x}(f) = \begin{cases} \sum_{n=-\infty}^{+\infty} \frac{1}{\eta} x\left(\frac{n}{\eta}\right) e^{2\pi j n \frac{f}{\eta}} & \text{if } f \leq \eta \\ 0 & \text{elsewhere} \end{cases} \tag{2.72}$$

2.2.3.2 Shannon sampling in multidimensional spaces

Now we will try to apply Shannon theorem, to multidimensional periodic signals in the general case.

Let x a continuous function of $\mathbb{L}^2(\mathbb{R}^n)$ such that the support of the Fourier transform of x is finite: $Supp\ \hat{x} \subset K \subset \mathbb{R}^n$ with K a bounded subset of $\mathbb{R}^n$.

Let W be a non-singular periodicity matrix, as seen in the section 2.2.2.3. Let's apply the same strategy used by Shannon, and defined a W^{-t}-periodized version of the Fourier transform of x:

$$\Gamma(\vec{f}) = \sum_{\vec{k} \in \mathbb{Z}^n} \hat{x}(\vec{f} - W^{-t}\vec{k}) \tag{2.73}$$

This new expression follows a W^{-t} periodic scheme in $\mathbb{R}^n$ then it can be approximated by a Fourier serie in n dimension, as defined in the section 2.2.2.3:

$$\Gamma(\vec{f}) = \sum_{\vec{k} \in \mathbb{Z}^n} c_{\vec{k}}(\Gamma) e^{2i\pi((W^{-t})^{-t}\vec{k})^{\intercal}\vec{f}} \tag{2.74}$$

$$= \sum_{\vec{k} \in \mathbb{Z}^n} c_{\vec{k}}(\Gamma) e^{2i\pi(W\vec{k})^{\intercal}\vec{f}} \tag{2.75}$$

with

$$c_{\vec{k}}(\Gamma) = \frac{1}{|\det(W^{-t})|}\langle \Gamma, E_{\vec{k}}\rangle_{\mathbb{L}^2_{P}(W^{-t})} \tag{2.76}$$

$$= \frac{1}{|\det(W^{-t})|}\int_{\mathbb{P}_{W^{-t}}} \Gamma(\vec{f}) e^{2i\pi(W\vec{k})^{\intercal}\vec{f}} \tag{2.77}$$

Under the condition that $\mathbb{P}_{W^{-t}}$ is a tiling of $\mathbb{R}^n$, and $K \subset \mathbb{P}_{W^{-t}}$ we have:

$$c_{-\vec{k}}(\Gamma) = |\det(W)| x(W\vec{k}) \tag{2.78}$$

Then, as seen in the 1-D version, we can derive the expression of the continuous fourier transform of x, from a discrete set of elements of x:

$$\Gamma(\vec{f}) = \sum_{\vec{k}\in\mathbb{Z}^n} \hat{x}(\vec{f} - W^{-t}\vec{k}) \tag{2.79}$$

$$= |\det(W)| \sum_{\vec{k}\in\mathbb{Z}^n} x(W\vec{k}) e^{2i\pi(W\vec{k})^{\intercal}\vec{f}} \tag{2.80}$$

The Shannon condition here, as seen previously correspond to $K \subset \mathbb{P}_{W^{-t}}$, or in other words, we must have

$$\Gamma(\vec{f}) = \hat{x}(\vec{f}) \quad \forall \vec{f} \in \mathbb{P}_{W^{-t}} \tag{2.81}$$

$$\chi_{\mathbb{P}_{W^{-t}}}\Gamma(\vec{f}) = \chi_K \Gamma(\vec{f}) = \hat{x}(\vec{f}) \tag{2.82}$$

With $\chi_{\mathbb{P}_{W^{-t}}}$ respectively χ_K the characteristic function of $\mathbb{P}_{W^{-t}}$ an respectively K
Now we can rewrite continuous expression of $\hat{x}$ and its inverse fourier transform x expressed from a discrete set of elements:

$$\hat{x}(\vec{f}) = \chi_{\mathbb{P}_{W^{-t}}} \sum_{\vec{k}\in\mathbb{Z}^n} \hat{x}(\vec{f} - W^{-t}\vec{k}) \tag{2.83}$$

$$x(\vec{t}) = |\det(W)| \sum_{\vec{k}\in\mathbb{Z}^n} x(W\vec{k}) \hat{\chi_{\mathbb{P}_{W^{-t}}}}^{-1}(W\vec{k} - \vec{t}) \tag{2.84}$$

Where $\hat{\chi_{\mathbb{P}_{W^{-t}}}}^{-1}$ is the inverse Fourier transform of the regular tile characteristic function. Actually, we can notice that when we have $K \subset \mathbb{P}_{W^{-t}}$, there is an infinity of interpolation function that give perfect estimation of x in any point of $\mathbb{R}^n$, of the form $\hat{\chi_L}$ for the Fourier transform of the characteristic function of any bounded set L that verify $K \subset L \subset \mathbb{P}_{W^{-t}}$

2.2.4 Efficient sampling for bandlimited function in a sphere

Let's now imagine that we have x a continuous function of $\mathbb{L}^2(\mathbb{R}^n)$ such that its Fourier transform has a finite support: $Supp \quad \hat{x} \subset K \subset \mathbb{R}^n$ with K a L^2 ball of radius b in $\mathbb{R}^n$. Now that we have defined a function bandlimited in an hypersphere of R^n, we will try to find the best possible sampling scheme, that will allow us to retrieve the whole function thanks to a discrete set of its samples.

This exercise is of interest in CBCT, because there is no physical apriori that tends to prove that signal of interest in reconstruction have anisotropic profiles in Fourier domain, hence the choice of a isotropic "friendly" sampling grid.

First, we know that we are looking for W a non-singular periodicity matrix such that we have:

$$K \subset \mathbb{P}_{W^{-t}} \tag{2.85}$$

Although it is not sufficient, we can translate this inclusion into a constraint over the hypervolume of the hyper-parallelogram $\mathbb{P}_{W^{-t}}$ that must be at superior or equal of that of the L^2 ball K of radius b: We can also say that if we want an efficient sampling, we want to maximise the sampling steps in the direct space, in order to reduce the sampling effort. In the fourier space, this is equivalent to reduce the distance between nodes, and we can translate this problem into a minimisation of the sampling lattice's primitive cell hypervolume, see [**middle1962sampling**] for a more exhaustive presentation of the problem. Here is one expression of the problem we want to solve for efficient sampling scheme:

$$\underset{W \in \mathbb{R}^{n \times n}}{min} |\det(W^{-t})| \quad \text{s.t} \quad |\det(W^{-t})| \geq V_n(b) \quad \text{and} \quad K \subseteq \mathbb{P}_{W^{-t}} \tag{2.86}$$

Where $V_n(b)$ is the expression of the volume of the L^2 ball of radius b in $\mathbb{R}^n$, that can be expressed separately, between even and odd dimensional cases:

$$V_{2k}(b) = \frac{\pi^k}{k!} b^{2k} \tag{2.87}$$

$$V_{2k+1}(b) = \frac{2(k!)(4\pi)^k}{(2k+1)!} b^{2k+1} \tag{2.88}$$

There are many existing proof of this formula, see [**wang2005volumes**] for more informations. We used this approach in order to help understanding the link between the sampling problem for n-dimensional bandlimited function, and the sphere packing problem, however, solving this problem in the general case, appeared to be a difficult task. Fortunately, [**middle1962sampling**] recall in his work that [**coxeter1951extreme**] gave quadratic form

that aimed to describe lattices vectors of optimal sphere packing lattices up to eight dimensions. Although these packings were not proved to be optimal at that time, these vectors will be sufficient for our work in 3 dimensions.

2.2.4.1 The Body Centered Cubic grid

In $\mathbb{R}^3$, Kepler conjectured in 1610 that the densest packing in 3D yielded a density of $\frac{\pi}{3\sqrt{2}} \approx 74.05\%$.
This conjecture was only proved in 1998, in [**hales1998kepler**].

Many different matrix can be derived for the same solution. The one we choose in the Fourier space could be described with the following periodicity matrix W^{-t}, in the Fourier domain :

$$W_{FCC} = W^{-t} = \begin{pmatrix} \sqrt{2}b & -\sqrt{2}b & \sqrt{2}b \\ \sqrt{2}b & 0 & -\sqrt{2}b \\ 0 & \sqrt{2}b & 0 \end{pmatrix} \tag{2.89}$$

with $|\det(W^{-t})| = 4\sqrt{(2)}b^3$, the volume of the parallelotope that represent the primitive cell of the periodic lattice. The packing density in this case reads:

$$\begin{aligned} \text{density} &= \frac{\text{sphere volume}}{\text{primitive cell volume}} && (2.90)\\ &= \frac{\frac{4}{3}\pi b^3}{4\sqrt{2}b^3} && (2.91)\\ &\approx 74.05\% && (2.92) \end{aligned}$$

It is interesting to derive the packing density for the cartesian lattice for comparison:

$$W_{CC}^{-t} = \begin{pmatrix} 2b & 0 & 0 \\ 0 & 2b & 0 \\ 0 & 0 & 2b \end{pmatrix} \tag{2.93}$$

Where W_{CC}^{-t} is the lattice matrix in Fourier space for spectrum periodization and $|\det(W_{CC}^{-t})| = 8b^3$ is the volume of the corresponding cube.

$$\begin{aligned} \text{density} &= \frac{\text{sphere volume}}{\text{primitive cell surface}} && (2.94)\\ &= \frac{\frac{4}{3}\pi b^3}{8b^3} && (2.95)\\ &\approx 52.36\% && (2.96) \end{aligned}$$

The density ratio of CC versus FCC is equal to $\frac{74.05}{52.36} \approx 1,41$ which imply that samples in fourier space will be packed 1.41 times more densely.
The primitive cell related to the grid described by the matrix presented here generates the so-called Face Centered Cubic lattice (FCC), whose Voronoï cell is the rhombic dodecahedron. The name "Face Centered" is easily understood looking at the figure 2.3, where we can see that the lattice nodes lies on the vertices and the center of the faces of a cube in $\mathbb{R}^3$.

Its reciprocal lattice can be expressed as:

$$W_{BCC} = W = \begin{pmatrix} \frac{1}{2\sqrt{2}b} & 0 & \frac{1}{2\sqrt{2}b} \\ \frac{1}{2\sqrt{2}b} & 0 & \frac{-1}{2\sqrt{2}b} \\ \frac{1}{2\sqrt{2}b} & \frac{1}{\sqrt{2}b} & \frac{1}{2\sqrt{2}b} \end{pmatrix} \tag{2.97}$$

Which generates the so-called Body Centered Cubic lattice (BCC), whose Voronoï cell is the truncated octahedron. Here again, the name "Body Centered" is easily understood looking at the figure 2.3, where we can see that the lattice nodes lies on the vertices and the center of the body of a cube in $\mathbb{R}^3$.
Its volume is equal to $|\det(W)| = \frac{1}{|\det(W^{-t})|} = \frac{1}{4\sqrt{2}b^3}$. It is interesting to derive the relative sampling density for the cartesian lattice for comparison:

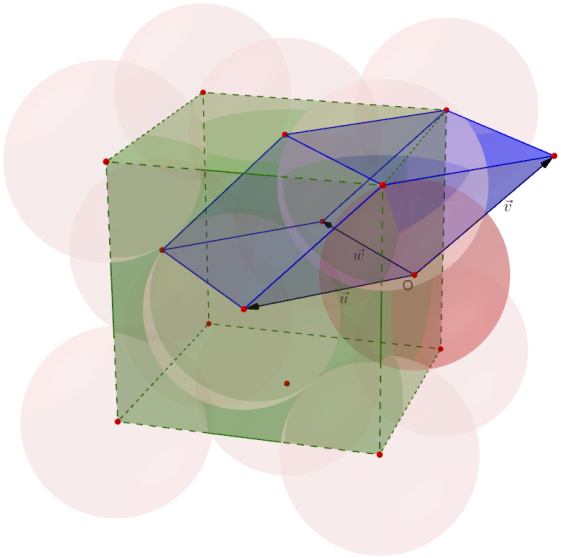
$$W_{CC} = \begin{pmatrix} \frac{1}{2b} & 0 & 0 \\ 0 & \frac{1}{2b} & 0 \\ 0 & 0 & \frac{1}{2b} \end{pmatrix} \tag{2.98}$$

Where and W_{CC} is the matrix lattice in direct space, with a primitive cell of volume $|det(W_{CC}^{-t})| = \frac{1}{|\det(W_{CC}^{-t})|} = \frac{1}{8b^3}$

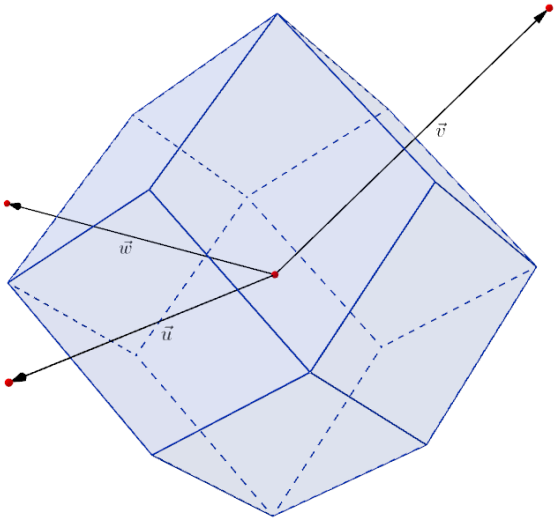
In direct space, the samples will be distributed more sparsely, with a ratio of $\frac{4\sqrt{2}b^3}{8b^3} \approx 0.71$ resulting in approximately 30% less samples on the direct grid keeping the same aliasing-free property in fourier space, thus allowing for a perfect Shannon reconstruction

Reconstruction/Interpolation in $\mathbb{R}^3$ As seen in the previous section, the function $\chi_{\hat{\mathbb{P}_{W^{-t}}}}^{-1}$ can be used in a convolution for interpolation in 3D, unfortunately the same drawbacks arising in *sinc* based interpolation applies in 3D, for instance the infinite support of this interpolation kernel.

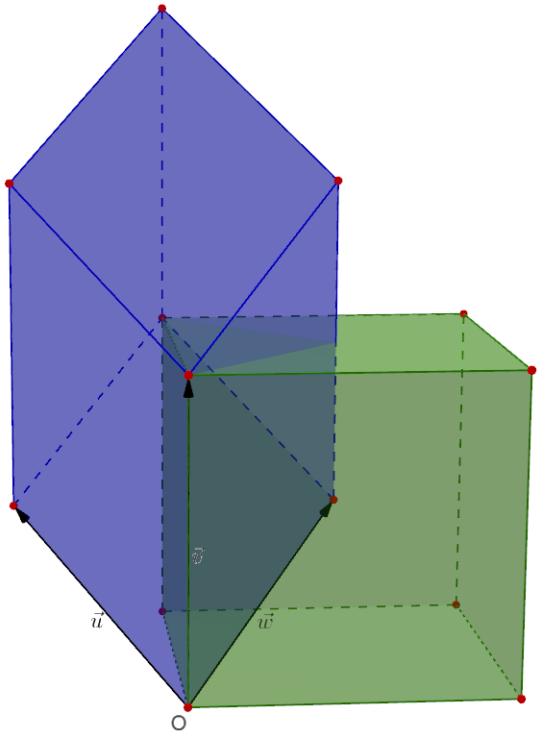
Numerous approaches have been derived in order to provide better interpolation kernels using those regular grids. Among the most successfull ones, we can cite the spline based interpolation, whose optimality property, even for non sphere-bandlimited signals have been studied in 2D in [**condat2005hexagonal**].



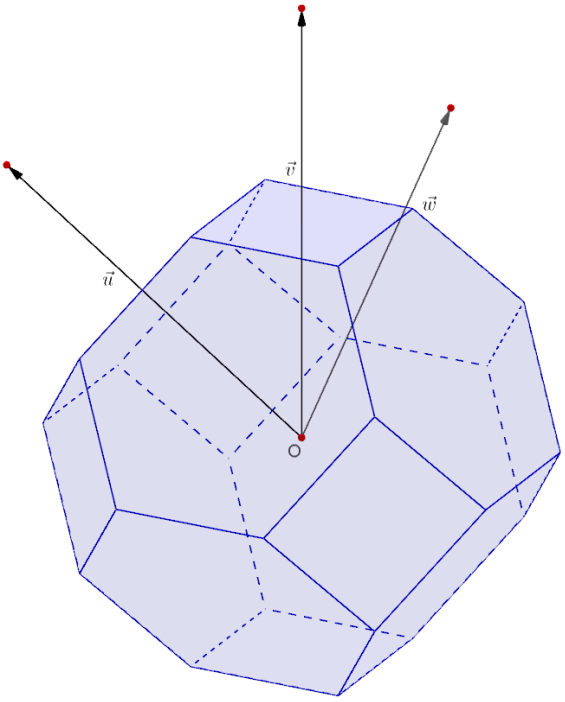
(a) FCC lattice: Fourier space



(b) FCC Voronoi cell: rhombic dodecahedron



(c) BCC lattice: direct space



(d) BCC Voronoi cell: truncated octahedron

Figure 2.3: Sampling lattices features in 3D

Concerning the spline based reconstruction, we have not found evidences that results obtained by Condat & Al on non sphere bandlimited signals apply to 3D box splines but some extension of those spline based interpolation to 3D for BCC and FCC grids have been investigated in [**entezari2008practical**], [**entezari2009quasi**], [**finkbeiner2010efficient**],

[**moren2012efficient**] and even more recently in [**schold2015image**].

2.2.4.2 Some properties of the body centered cubic lattice

Isotropy of the BCC Voronoi cell An interesting study on the use of the BCC for 3D tomographic reconstruction in [**mueller2009optimal**] previously highlighted the better isotropy of BCC Voronoi cell, regarding the cube for CC and the rhombic dodecaedron for the FCC. We decided here to give a more explicit comparison of the surface of various geometric parallelotope, including the cube, where all volume are normalized to 1:

- Cube:
  Surface $= 6 \times a^2$
  volume $= a^3$ with a the length of an edge
  Surface of the unit volume:

$$= 6 \times \left(\frac{1}{\sqrt[3]{1^3}} \right)^2 \tag{2.99}$$

$$= 6 \tag{2.100}$$

- Rhombic dodecahedron:
  Surface $= 8\sqrt{2}a^2$
  volume $= \frac{16}{9}\sqrt{3}a^3$ with a the length of an edge
  Surface of the unit volume:

$$= 8\sqrt{2} \left(\frac{1}{\sqrt[3]{\frac{16}{9}\sqrt{3}}} \right)^2 \tag{2.101}$$

$$= 5.34 \tag{2.102}$$

- Truncated octaedron:
  Surface $= (6 + 12\sqrt{3})a^2$
  volume $= 8\sqrt{2}a^3$ with a the length of an edge
  Surface of the unit volume:

$$= (6 + 12\sqrt{3}) \left(\frac{1}{\sqrt[3]{8\sqrt{2}}} \right)^2 \tag{2.103}$$

$$= 5.31 \tag{2.104}$$

- Sphere:
  Surface $= 4\pi r^2$
  volume $= \frac{4}{3}\pi r^3$ with r the radius of the sphere

Surface of the unit volume:

$$= 4\pi \left(\frac{1}{\sqrt[3]{\frac{4}{3}\pi}} \right)^2 \tag{2.105}$$

$$= 4.83 \tag{2.106}$$

We can see that the truncated octahedron can pack more volume with less surface, and then has a form closer to a sphere, yielding a better isotropy.

Neighborhood in the BCC lattice and interpolation As stated in [**entezari2008practical**], and as presented previously, the Voronoï cell of the BCC lattice is a truncated octahedron. We can also consider the parallelotope where each vertex correspond to one of the first immediate neighbors of a lattice point. This structure can be found by Delaunay tetrahedralization, where each point q is a first neighbor of p if their respective Voronoï cell share a non-degenerate face.
Using this definition, we can find 14 first neighbors that generates a rhombic dodecahedron:

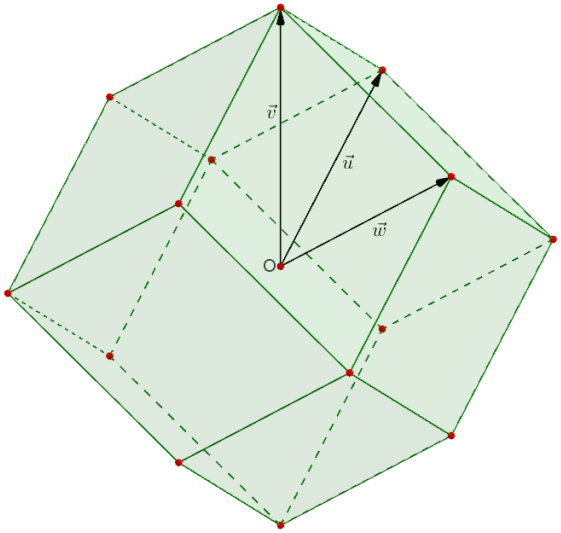


Figure 2.4: BCC lattice:
rhombic dodecahedron using 14 first neighbors

Entezari & al, in [**entezari2008practical**] and [**entezari2009quasi**] exploited this geometrical structure in order to form advanced interpolation kernel based on 3D Box-Splines that we will be using in the section 3.3.

2.2.5 Cartesian Sampling in 3D

Although it does not feature optimality property for functions bandlimited in a sphere, the most common grid used in engineering for CT reconstruction is the cartesian grid, and is

mathematically defined through a simple diagonal 3×3 matrix M_{cart}, called, as seen earlier, the lattice matrix, where each column defines a periodization vector.

$$M_{cart} = (\vec{u}, \vec{v}, \vec{w}) = \begin{pmatrix} tx & 0 & 0 \\ 0 & ty & 0 \\ 0 & 0 & tz \end{pmatrix} \tag{2.107}$$

In this case, the nodes of the grid are defined by all possible integer linear combination of the periodization vectors, this topic has already been studied in depth in section 2.2.2 along with alternative regular grids.

The volume elements in this case, are the Voronoï cells of this grid: which are axis aligned rectangular cuboid. We will use this model throughout this work, unless stated otherwise.

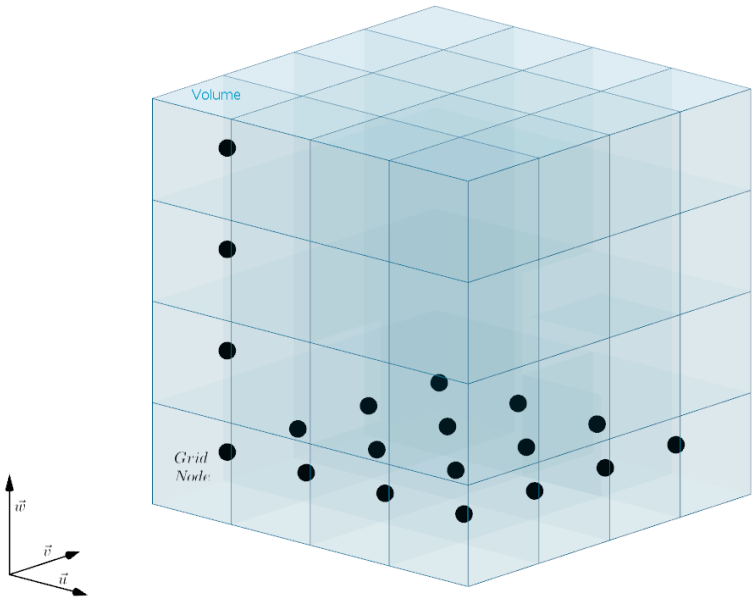


Figure 2.5: Volume discretization using a Cartesian grid

2.2.6 Integral sampling process

In the case of integral measurments that arise in transmission tomography, the discretized attenuation map μ_{dis}, can be seen as a continuous attenuation function $\mu(\vec{t})$, undergoing a two step transformation, that has the following definition:

- A convolution of the function $\mu(\vec{t})$ over $\mathbb{R}^3$, by the function $\chi_0(\vec{t})$, which is the voxel indicator, or voxel function related to the grid node of index 0 of coordinates $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$

- Multiplication in direct space with the grid sampling operator $S_{M_{grid}}(\vec{t}) = \sum_{\vec{k}\in\mathbb{Z}^n} \delta\left(\vec{t} - M_{grid}\vec{k}\right)$ With δ the Dirac distribution.

Following the definition of sampling given in equation 2.2.6, we can see 3 potential drawbacks, that will guide our discretization strategy:

2.2.6.1 Spectral point of view

The convolution with a non isotropic volume element indicator that occurs during the sampling scheme, can be analysed from a spectral point of view. Using the convolution theorem, this convolution can be seen as a multiplication with the Fourier transform of the voxel indicator function in Fourier space.

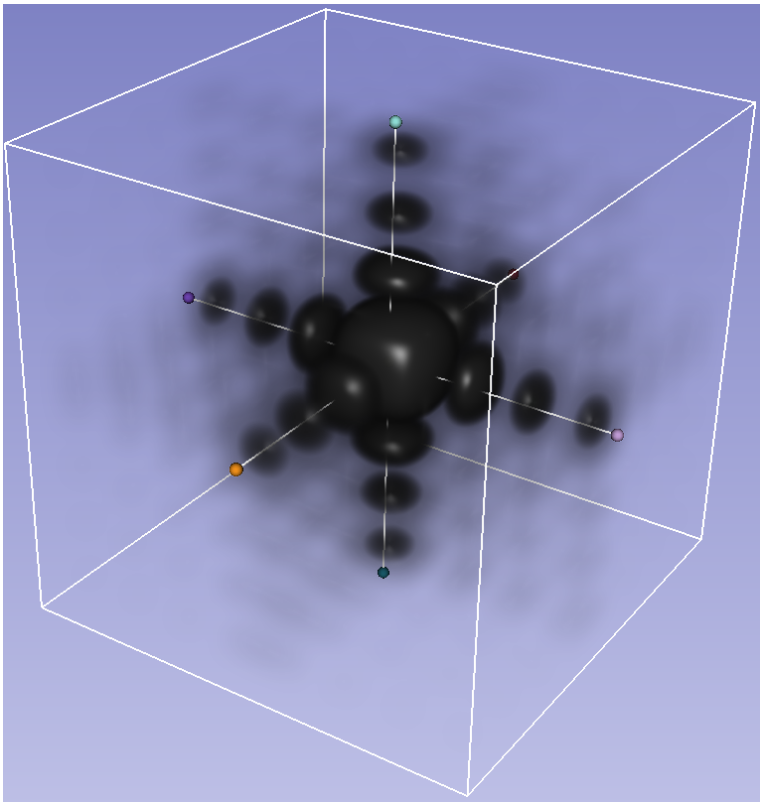


Figure 2.6: Rendering of the 3D DFT of a voxel indicator function

Unfortunately, as seen on figure 2.6, the Fourier transform of the cartesian grid canonical voxel function, which is a tensor product of cardinal sine functions, has an anistotropic profile, which tends to exhibit more high frequencies along axis.

The sampling step, that comes after the convolution, can be analysed using Shannon theorem. The multidimensional Shannon theory exposed in section 2.2.3, explains how aliasing

cancellation property, in the case of general grids, rely on geometrical intersections between spectral support.

In the case of cartesian grid, Shannon condition holds when considering spectral cubes of data, such that any signal whose spectrum lies inside the Voronoi cell of the dual grid, here, the cube of size $\left(\frac{1}{2tx}, \frac{1}{2ty}, \frac{1}{2tz}\right)$ is aliasing free. Unfortunately, as seen previously, as voxel indicator does not feature a perfect frequency cutoff behaviour, the previous convolution process does not ensures that the aliasing property holds independently of the function $\mu(x)$, especially for axis aligned signals. Aliasing-free sampling condition then reduces to Shannong conditions over $\mu(\vec{t})$, hence relying on physical apriori about the spectrum support of the attenuation function $\mu(x)$.

We can retain that aliasing can be reduced using 2 strategies:

- Increasing the grid resolution, or equivalently minimizing the determinant of the corresponding lattice matrix, so that the dual grid Voronoi cell match the physical signal spectral support
- Replace the voxel function, from the direct grid Voronoi cell indicator function to a smoother function exhibiting a proper antialiasing behaviour through high frequency cutoff filtering.

2.2.6.2 Physical point of view

From a physical point of view, the discretization of the space leads to the so-called partial-volume effects. The fact that voxels samples may account for a mixture of material with different density restrain the definition of small structures, and hinder interpretation of CT data, although some strategy based on apriori knowledge can be used to derive simple super-resolution reconstruction, see for instance [**wellington1987x**].

The previous remark also tends to favor discretization based on high resolution grids.

2.2.6.3 Algebraic point of view

Independently of the method used to formalize the reconstruction problem, the input data will be the X-ray detector pixels intensity, and, the expected output will be the correct attenuation value for each voxel, that fit the projection data using the current model.

Unfortunately, the higher is the grid resolution, the worst the known/unknown ratio get, because no additional information is added about the projection data. We will elaborate more on this topic in chapter 4.

2.2.7 Spherically symmetric volume elements

As seen in , Voronoi cell indicator functions of most of the regular grids in the direct domain does not exhibit a good aliasing filtering due to their lack of regularity.

One can however think about replacing such functions, by smoother functions with higher regularity defined for each grid node. Defining such functions amounts to extend the field of filter design to the multi-dimensional case, where the difficulty is increased because regular grids are not as isotropic as in 1 dimension. Plus, in order to be used in tomographic imaging models, one should be able to compute integrals along lines through these 3D filters.

One of the best tradeoff that has been found is the use of low pass spherically or radially symmetric functions, whose Abel transform is known. The fact that those spherically symmetric functions have a spherically symmetric spectral profile, decreasing in a radially symmetric manner also allows to use even more efficiently the BCC and FCC grids presented earlier.

Let's review some of the most reknown radial basis functions in use in tomography

2.2.7.1 Kaiser-Bessel blob

Prolate spheroidal wave function Prolate spheroidal wave function are a set of functions defined from a specific combinaison of windowing functions. Assuming we want to window a signal in direct space such that its support is $[-\frac{T}{2}, \frac{T}{2}]$, we can define the rectangular windowing operator R_T such that, for every square integrable function $x \in L^2(\mathbb{R})$ we have $supp R_T\ x \quad \in [-\frac{T}{2}, \frac{T}{2}]$. This linear windowing operator in direct space, is then followed by another windowing operator L_F, which aims at performing a rectangular windowing function in the Fourier domain, equivalent to a perfect low pass filter, defined as $supp L_F\ \hat{x} \quad \in [-F, F]$ with $\hat{x}$ the Fourier transform of x. Lastly, the direct space windowing operator is applied to the output of the low pass filter, which result in the composite operator $R_T L_F R_T$ which has the nice property of being bounded, and self-adjoint.

Using a general version of the spectral theorem to bounded symmetric operators, we can derive a non-empty set of orthogonal eigenfunctions ψ_n such that $R_T L_F R_T \psi_n = \lambda \psi_n$.

The family of functions ψ_n that have a finite support in direct space are called the PSWF[5], they have been studied in [**slepian1961prolate**], along with the uncertainty principle in the framework of signal processing.

Kaiser-Bessel windowing function In [**harris1978use**], the author studied the problem of optimal windowing along with the concept of uncertainty, that intrinsically forbid a perfect selectivity in both time and frequency. The author study multiple metrics, like the optimal time-bandwidth product, the finite support function that minimize the the main lobe width for a given sidelobe level, and the finite support function that maximizes the energy

[5] *Prolate Spheroidal Wave Functions*

in the band of frequencies $[-F, F]$. He recall that the later criteria was optimized by the PSWF of order 0, which where parametrized by the time-bandwidth product, and that Kaiser in [**kaiser1966system**] gave a simple approximation of these function using the zero-order modified Bessel function of the first kind.

This function has one parameter α that allows to tune for the regularity/selectivity tradeof between direct and Fourier space, and is defined as follows:

$$K(a, \alpha, r) = \begin{cases} \frac{I_0[\pi\alpha\sqrt{1-\left(\frac{r}{a}\right)^2}]}{I_0(\pi\alpha)} & \text{if } 0 \leq r \leq a \\ 0 & \text{otherwise} \end{cases} \tag{2.108}$$

Where we have:

- I_0 stands for the modified Bessel function of the first kind order 0
- a is the radius of the support of the function
- $\pi\alpha$ is half of the time-bandwidth product. It can be understood as a spatial selectivity parameter, or the inverse of a smoothness parameter, the higher it gets, the better is the spatial selectivity, but the widest is the corresponding spectral window, hence lowering the spectral selectivity and the lowpass filtering efficiency.
- r is the distance from the origin, where the function is evaluated

Generalized Kaiser-Bessel function for CT Kaiser-Bessel windowing function has been studied in the specific framework of multidimensional CT imaging in [**lewitt1990multidimensional**], and the author gave a generalized version of the expression of its profile:

$$P_{KB}(m, a, \alpha, r) = \begin{cases} \frac{[\sqrt{1-\left(\frac{r}{a}\right)^2}]^m I_m[\alpha\sqrt{1-\left(\frac{r}{a}\right)^2}]}{I_m(\alpha)} & \text{if } 0 \leq r \leq a \\ 0 & \text{otherwise} \end{cases} \tag{2.109}$$

along with the expression of its Fourier transform:

$$\hat{P}_{KB}(m, n, a, \alpha, r) = \begin{cases} \frac{(2\pi)^{n/2} a^n \alpha^m}{I_m(\alpha)} \frac{I_{n/2+m}[\sqrt{\alpha^2-(2\pi ar)^2}]}{[\sqrt{\alpha^2-(2\pi ar)^2}]^{n/2+m}} & \text{if } 2\pi ar \leq \alpha \\ \frac{(2\pi)^{n/2} a^n \alpha^m}{I_m(\alpha)} \frac{J_{n/2+m}[\sqrt{(2\pi ar)^2-\alpha^2}]}{[\sqrt{(2\pi ar)^2-\alpha^2}]^{n/2+m}} & \text{if } 2\pi ar \geq \alpha \end{cases} \tag{2.110}$$

Where n is the dimension of the euclidean space we are working in, which is 3 in our the framework of our study.

The autho also derived a closed form expression of the Abel transform of K:

$$A_{KB}(m,a,\alpha,r) = \begin{cases} \frac{\alpha}{I_m(\alpha)}\sqrt{\frac{2\pi}{\alpha}}[\sqrt{1-\left(\frac{r}{a}\right)^2}]^{m+1/2} I_{m+1/2}[\alpha\sqrt{1-\left(\frac{r}{a}\right)^2}] & \text{if } 0 \leq r \leq a \\ 0 & \text{otherwise} \end{cases} \quad (2.111)$$

Kaiser-Bessel function is the most widely used RBF[6] for CT, its use has been reported in [**lewitt1992alternatives**], [**matej1995efficient**] and [**ziegler2006efficient**]

2.2.7.2 Gaussian blob

Heisenberg-Pauli-Weyl inequality The Heisenberg-Pauli-Weyl inequality reads:

$$\left(\int_{\mathbb{R}^n} |\vec{t}|^2 |x(\vec{t})|^2 d\vec{t}\right)\left(\int_{\mathbb{R}^n} |\vec{f}|^2 |\hat{x}(\vec{f})|^2 d\vec{f}\right) \geq \frac{\|x\|_2^4}{16\pi^2} \quad (2.112)$$

Where x is a square integrable function: $x \in L^2(\mathbb{R})$, and C is a constant from $\mathbb{R}$. The history of this inequality as well as its generalization to various power of $|\vec{t}|$ and $|\vec{f}|$ and other L^P spaces has been studied in [**folland1997uncertainty**].

This remarkable inequality establish a theoretical bound regarding the product of the spread of a signal in time domain, and its spread in Fourier domain, where the spread is measured in terms of the order 2 moment. A simple conclusion we can drive from this inequality, is that, we cannot design a filter that features a short support and a good frequency selectivity in the mean time. Moreover, it is easy to show that a particular function reach the optimal tradeof, namely the Gaussian function.

Gaussian blob for CT The use of Gaussian blob remained more confidential in CT reconstruction, we can cite the work of Hanson and Wecksung [**hanson1985local**] and more recently Wang & Al in [**wang2011image**].

Let's define this radial volume element, such that its integral over $\mathbb{R}^3$ is normalized to 1:

$$P_G(r,\alpha) = \frac{1}{\alpha^3\sqrt{\pi^3}} e^{-\frac{r^2}{\alpha^2}} \quad (2.113)$$

And its Abel transform reads

$$A_G(\alpha,r) = \frac{1}{\alpha^2\sqrt{\pi^2}} e^{-\frac{r^2}{\alpha^2}} \quad (2.114)$$

[6] *Radial Basis Function*

Unfortunately, although gaussian family functions are part of the Schwartz space, ie, it is a rapidly decreasing function, its support is infinite. Regardless of how gaussian blob-based will be implemented, it should always be analysed as a truncated gaussian, hence loosing its appealing theoretical optimality properties. The radial truncation in direct space can be analysed as a multiplication with a radially symmetric analog of the rectangle function, sometimes called *circ* and its Fourier transform that is a radially symmetric analog of the cardinal sine, based on Bessel functions, sometimes called *jinc*, but this topic is beyond the scope of our study.

2.2.7.3 Mexican Hat

To our knowledge, the use of bandpass blobs for CT reconstruction have only been described in [**wang2011methodes**]. In this work, the author describe a multiple band and multiscale approach to image representation based on smooth RBF.

Among the multiple bandpass functions that have been described in this work, we found the Mexican Hat blob to be one of the most interesting due to its simplicity, and numerical efficiency. The mexican hat function encountered a large success in the field of computer vision, geosciences, it has been used to define the Ricker wavelets, and is simply defined as the second derivative of a gaussian.

This volume elements reads

$$P_{MH}(r,\alpha) = \frac{1}{\alpha\sqrt{\pi}}\left(r^2 - \frac{1}{2}\alpha^2\right)e^{-\frac{r^2}{\alpha^2}} \tag{2.115}$$

And its Abel transform reads:

$$A_{MH}(r,\alpha) = r^2 e^{-\frac{r^2}{\alpha^2}} \tag{2.116}$$

As this volume element, when considered in 1D has two vanishing moments, and is known to be a base component of a multiscale wavelet system, see [**daubechies1992ten**], it could potentially allow for a composite representation model, see the work of Han in [**wang2011methodes**] for more relevant informations about this topic.

2.3 Conclusion

In this chapter, we introduced the topic of image formation in X-Ray transmission tomography, presented the simplified model we choose in this thesis, and justified our choice with simple statistical consideration. We also introduced various discretization scheme, and the related

issues, from a signal processing perspective. In the next chapter, we will see how those models can be used in the framework of CBCT imaging, from a practical point of view.

Chapter 3

Chapter 3: High performance implementation of tomographic operators

Sommaire

3.1 Introduction

In this chapter, we propose to study the design of tomographic operators in 3D cone beam geometry with flat panel detector, and give some elements related to their implementation in the framework of high performance computing with GPU.

The study will also recall some elements of projective geometry, that will be used throughout the chapter in order to derive generic tomographic operators that should be valid for arbitrary cone beam geometry, whithout any axis alignment apriori.

Then we will propose a simple framework to study and implement a blob based projector, compliant with arbitrary cone beam geometry, and arbitrary volume discretization scheme.

3.2 High performance computing with GPUs

3.2.1 A bit of history

GPU is the acronym that stands for graphics processing unit, refers to a computing hardware that was historically designed to execute in an efficient way the classical rendering pipeline, outputing images to a frame buffer, that inteded to be printed out on a screen.

Although we can date the first use of GPU with the design of arcade system boards to the 1970's, the first real chip that were able to execute a set of instructions dedicated to graphical rendering of 2D or 3D geometric objects were designed in the 1980's.

Since then, GPU became a one of the key element of the video gaming industry, allowing developers to design increasingly more realistic rendering engines. Although, our tomographic model can be interpreted as a specific case of rendering operation, in this thesis, we wil not be interested in the prebuilt libraries dedicated to this task, instead, we will focus on GPGPU for General-purpose computing on graphics processing units.

3.2.1.1 General-purpose computing on GPUs

Until the mid 1980's, the architecture of GPU, conversely to the one of x86 processors only allowed a small set of instructions often including low precision fixed point arithmetic for texture mapping, and simple memory manipulation. However, it is interesting to see that, even in this challenging envirronment, the first reported use of GPGPU for scientific computing in [**larsen2001fast**] used texture maps, limited to 8-bit precision for their matrix-matrix multiplication.

According to [**du2012cuda**], the first GPU to feature floating point unit did not arrived until 2003, but since then, the success of 3D video game engines, and their increasing demand

in computing power and memory bandwidth for rendering, illumination, texture mapping drove the high end GPU market.

Although GPUs started to become powerful floating point coprocessors on personal computers, their programmability remained poor, and mainly restricted to the interaction with classical graphical rendering libraries like OpenGL or DirectX.

The first devlopment kit, fully compatible with GPGPU programming was released by NVidia in the beginning of 2007, and it was the beginning of a new era for scientific computing.

3.2.2 Specific hardware capability

Although there are various GPU chip architectures, we will focus in this thesis on a class of CUDA-capable architectures, proposed by NVidia, including Kepler, Maxwell, and Pascal, with compute capability ranging from 3.0 to 6.0. An overview of the architecture of the GP100 Chip can be found on the figure 3.1.

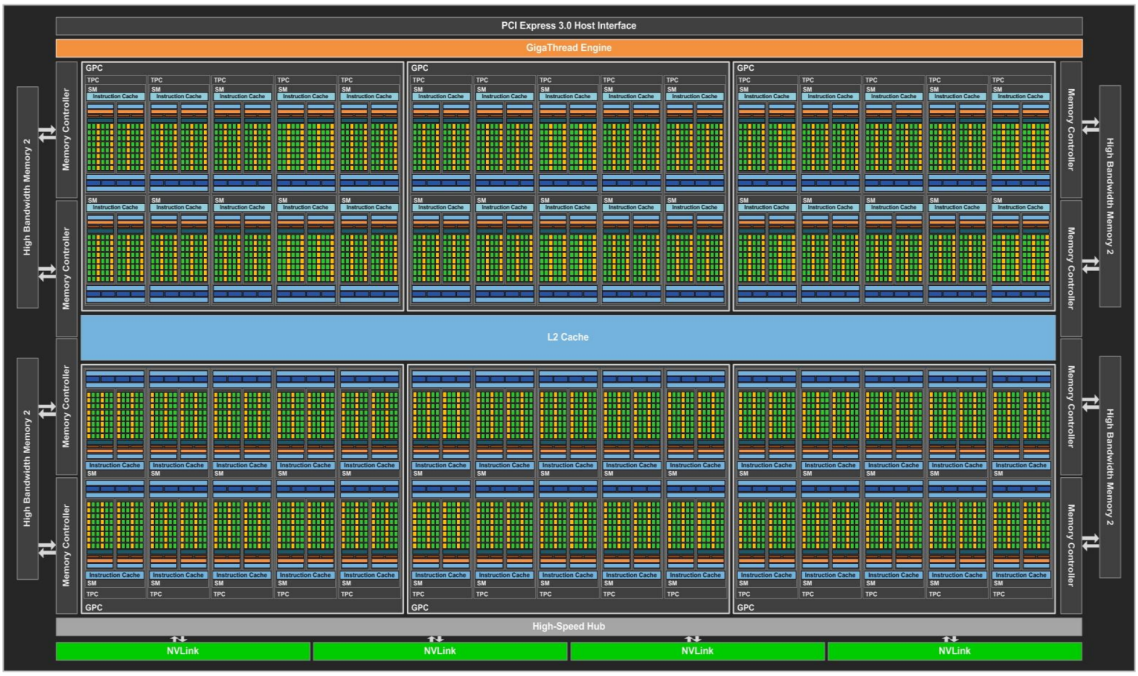


Figure 3.1: GP100 Chip architecture, Courtesy: NVidia

On the figure 3.1, we can see some specific hardware capabilities that we will be interested in the framework of high performance tomographic reconstruction:

- **The HBM2 memory:** in 2016, this recent evolution of the GDDR5 memory offers a extremely high bandwidth when coupled to a large memory bus of 4096 links: up to 732 GB/s (data: NVidia). This value represents approximately a ten fold improvment

over the 76.8 GB/s of theoretical bandwidh the intel E5-2699V4 processor offers when equipped with DDR4 SDRAM (data: Intel).

- **The L2 Cache:** This element is of central importance in the memory hierarchy model of modern computers. The GP100 has 4096 KB of unified L2 Cache, meaning that plays the role of data cache, as well as instruction cache. This value compares less favorably with the architecture of most modern CPU, like the Intel E5-2699V4, this later one having a L2-L3 hybrid caching policy called smart-cache, based on a total memory capability of 55 MB.

- **The Giga-Thread Engine:** Aside from its tremendous memory bandwidth, the CUDA-capable architectures offers a hardware-based thread engine. It means that thread context creation, ordonnancing, and instruction dispatch, that are managed in blocks at this level, and by warp at the SM[1] level, can be handled in an extremely fast manner, by a hardware component. Actually, the definition of thread in the world of GPU computing differs greatly from the software defined thread managed by operating systems. It can be seen as a single instance of kernel code execution in the framework of GPU computing, but we will give more details about this model later.

- **The Streaming Multiprocessors:** The blocks of threads managed by the Giga-Thread Engine are dispatched to the multiprocessors, to be executed. One block of thread can be attached to only one SM, but one SM can host up to 32 blocks simultaneously, within a limit dictated by resource consumption per thread (like register and shared memory). The more SM there is per chip, the more parallel thread can be run in parallel. The GP100 counts 56 SM, and blocks are limited to 1024 threads, it could be tempting to deduce that the maximum number of thread that can be run in parallel is $56 \times 32 \times 1024 = 1,835,008$ but of course, it is very unlikely that a SM is able to execute 32 full blocks of 1024 threads.

We have seen that the SM are hardware components of significant importance in the architecture of NVidia GPUs, let's now take a closer look at their structure, presented in figure 3.2.

Here again, we will highlight some architecure specificity that we will try to leverage while implementing reconstruction algorithms :

- **Shared memory,Texture,L1 cache:** After the registers, these storage banks are the most efficient ones, both in terms of bandwidth and latency. Although, registers/L1 and L2 cache memory hierarchy are nearly transparents for the programmer, unless specific compilation flags are used, shared memory and texture cache capabilities must be used explicitly in order to be leveraged. Shared memory can be dynamically allocated in the cuda programming language, and its interest lies in the fact that all thread from one block can read and write from it, allowing for fast communication, in the case of

[1] *Streaming Multiprocessor*

Figure 3.2: Streaming Multiprocessor for the Pascal architecture, Courtesy: NVidia

reduction for instance. The interested reader can refer to the surface and texture object API, as well as the shared memory in the NVidia documentation.

- **Texture units:** When using specific types of memory layouts called cuda arrays, it is possible to specify the intrinsic dimension of the data in order to favor data locality aware caching policies. One of the role of texture units is to manage memory transactions with the texture cache, this task include addressing, and filtering. The hardware acceleration of texture filtering, in particular linear, bilinear and trilinear texture filtering represents an important asset for multiple signal processing related softwares.

- **Warp scheduler and dispatch unit:** Inside the SMs, thread can only be run in a SIMD[2] manner that is somehow similar to the concept of vectorization for CPUs. The group of threads that will be working in parallel on one vector is called a warp, and its cardinality is equal to 32 on current NVidia GPUs. Warp scheduler and dispatch unit are in charge of managing the warp execution workflow, so that the processing pipeline utilization is optimized, and the occupancy of the SM is maximized. It should be noticed that in the framework of CUDA computing, the occupancy stands for the ratio between the actual number of warps running simultaneously and the maximum number of warps that can theoretically run in parallel onto one SM.

- **CUDA Cores:** Cuda cores can be basically understood as being equivalents of ALU[3] / FPU[4] on modern CPUs, so that it would be wrong to compare one CUDA core from NVidia architecture, to one Intel CPU core, that can be far more complex. As per 2016, the latest Pascal architecture SMs embed 64 cuda cores, which adds up to 3584 cores for a single card. A high end server CPU like the Intel E5-2699V4 embeds 22 cores, that can

[2] *Single Instruction Multiple Data*
[3] *Arithmetic Logic Unit*
[4] *Floating Point Unit*

use the hyperthreading technology to run 44 threads in parallel, at a faster rate: 3.60 GHz versus 1480 MHz for the GP100. The large discrepancy between these numbers indicates that one of the architecture will probably be more efficient at performing a massive amount of floating point computations

- **DP unit, SFU:** The double precision units are simply FPUs dedicated to double precision floating point operand. Special function units are hardware component that offers accelerated implementation of specific mathematical function like cosinus, exponential, etc...

3.2.2.1 Exploring the memory hierarchy in CUDA

When profiling a CUDA code, it may be interesting to measure metrics related to the computing architecture presented in the beginning of section 3.2.2. For instance harvesting metrics on the number of load/store and fp32 operations may allow to derive the arithmetic intensity of a kernel. Similarly, knowing the ratio of cache-hit may allow to check if data locality or the use of texture is relevant. We figured a simplified model of cuda memory hierarchy, along with their nvprof debugger keywords, on figure 3.3.

3.2.3 Programmability

Most of the code developed during this thesis was written in C++, which is a multi-paradigm language.

3.2.3.1 Modern C++ in gpu computing

There are many DSL that have been developed in order to leverage the computational power of GPUs but most of them rely on two languages/APIs: OpenCL, which is an open standard, and CUDA, a proprietary solution.

Although both language have their strenght and weaknesses, we found that CUDA provided a better expressivity, and most importantly, a better compatibility with modern standards of C++, like C++11, and probably c++14 in the future. For instance, the fact that CUDA kernels can be templated, and do support variadic templates helps to build more generic code and leverage compiler optimization capabilities.

During this thesis, we also extensively used a template-only library called *Thrust*, part of the cuda toolkit, that we found very powerful because of its ability to enable writing code following the functional paradigm.

Here is a summary of the functional concept that we used in cuda, or in c++ only code, with examples:

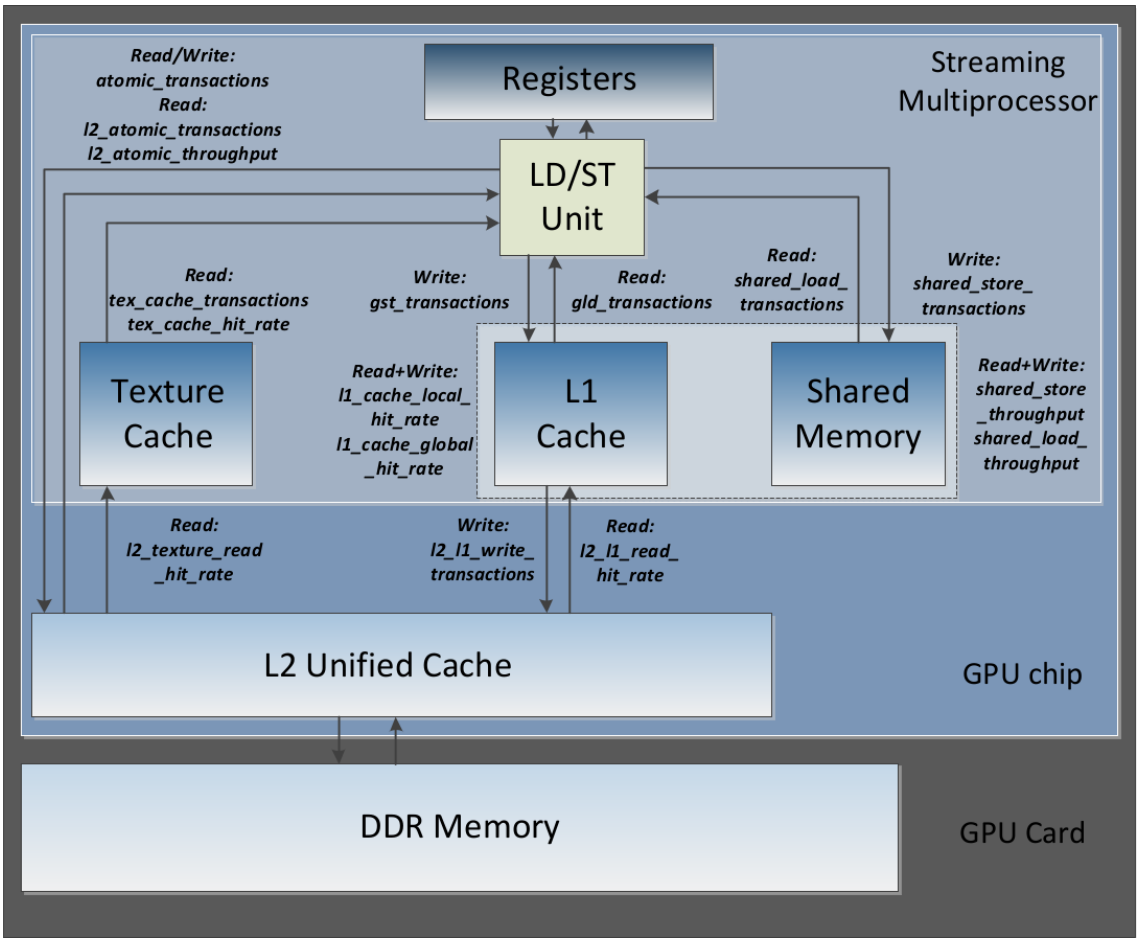


Figure 3.3: Memory hierarchy on NVidia GPUs, along with their profiler keywords

- **closure** A closure is a function that is able to embed some variables from the local context where it has been defined. The introduction of lambda functions in C++11 allowed to simulate a closure by using the concept of variable capture, that lowers the barriers imposed by differentiated host and device scope. The fact that cuda supports c++11 lambda functions allows to ease programming by limiting explicit kernels parameters definition.

- **higher order functions** Although at the time those lines are written, cuda 8 toolkit does not support officially C++14, it is worth mentionning that this C++ standard made the design of higher order functions easier. We figured an example in table 3.2.3.1 with a composition function. In this famous example, we can see that one can define recursively a "comp" function with a variadic template that allows an arbitrary large number of parameters, which are themselves functions. The comp function returns a function, which represents the composition of all functions passed as parameters, without making any apriori on their signature, all compliance checks are actually performed by the compiler at compile-time.

```
#include <iostream>

template<typename T0, typename... Tn>
auto comp( T0 f0, Tn... fn) {
  return [f0,fn...](auto a){ return f0(comp(fn...)(a)); };
}

template<typename T0>
auto comp( T0 f0 ) {
  return [f0](auto a){ return f0(a); };
}

auto square = [](auto n) { return n*n; };

int main(int argc, char* argv[]) {
  int t = 3;
  auto square2 = comp(square,square);
  std::cout << "Square of square of "<<t<<" is "<<square2(t)<<
     std::endl;
  return EXIT_SUCCESS;
}
```

- **Map** Operator mapping over a range is a very common concept in functional programming lanquages. The critical importance of this concept in the mapreduce pattern, popularized by Hadoop and Apache Spark technologies recently attracted a lot of attention, hence it is increasingly used by software developers from many fields. *Thrust* implements multiple flavours of operator mapping, among which the most generic is probably *thrust::transform*, that was designed to map any kind of unary or binary operator over 1 or 2 input range, and one output range.

- **Reduce** Reduction based on associative operator is also a very iconic pattern in functional programming, and has been extensively studied in the litterature of parallel computing. However, although implementing this pattern in C++ using thrust may seem straightforward, one must be warned about the common pitfall encountered in data reduction. When dealing with floating point operand, it should be noticed that most of the classical arithmetic operators are generally not associative. The non-associativity result in part from the fact that floating point operations may overflow, which can usually be detected at runtime, but the other aspect is more insidious: due to the finite size of their significand, arithmetic operator may silently convert small operands to zero, resulting in large errors in the final reduction. The interested reader may refer to [**ieee2008754**] and [**whitehead2011precision**] for detailed informations about floating point standard. To our knowledge, there is no cuda library that features error compensation method, like Kahan summation algorithm, see [**robey2011search**], however, the *thrust::reduce* function should allow one to define its custom reduction operator, and setup the Kahan summation algorithm by hand. It should also be noticed that *Thrust* library assumes

that all reduction operators are commutative. Non-commutative reduction operator should be used through *thrust::inclusive_scan*.

- **Lazy evaluation** This concept is also related to functional programming models, it allows to define objects of interest as a specific combination of functions, without requiring any evaluation unless a side effect demands it. This concept is not a builtin feature of C++, nor cuda, but it can be set up relatively easily to a limited extent by the mean of *thrust::make_transform_iterator*. Objects of this class can be simply constructed by using an input iterator, and an arbitrary operator, consistent with the input iterator. In this framework, we can define a composition of transform_iterators of arbitrary depth, without requiring any calculation unless the transform iterator is dereferenced at runtime, hence the concept of lazy evaluation. We present on table 3.2.3.1 a simple exemple of this concept, that allows to compute any arbitrary long sum, of the sequence $u_n = 2n + 1$, while using counting iterator, which are "virtual" iterators that does not require storage space.

```
#include <thrust/iterator/transform_iterator.h>
#include <thrust/iterator/counting_iterator.h>
#include <thrust/reduce.h>
#include <iostream>

int main(void) {
  //Declaration and initialization of a Host vector
  thrust::counting_iterator<int> beg(1);

  //Define a transform operator that multiply by 2, then add 1
  auto doubleIterator = thrust::make_transform_iterator( beg,
    [] __host__ __device__ (decltype(*beg) a){return 2*a;});
  auto doublePlusOneIterator = thrust::make_transform_iterator(
      doubleIterator,
    [] __host__ __device__ (decltype(*doubleIterator) a){return a
      +1;});

  // Make a reduction over a range of transformed iterators
  std::cout << "Sum of the first 10 elements of the sequence is
     "
    << thrust::reduce(doubleIterator,doubleIterator+10) <<std::
       endl;

  return(EXIT_SUCCESS);
}
```

More generally this kind of design, based on the concept of lazy evaluation is of critical importance in cuda, because operator to data mapping on the GPU implies a lot of interaction between the host and the GPU through cuda API, some synchronization, and more important, each kernel instance will require memory load and store operation.

Compile-time operator fusion through the use of transform iterators allows the compiler to refactor the various functors defined in the transform iterator into a single kernel, which is equivalent to operator composition, with a minimum of actual load and store operations, while keeping a loosely coupled design for the functors.

- **zip** Although it is not a critical concept in functional programming, the use of zip operator, through *thrust::make_zip_iterator*, can be used to design virtual structures of arrays (SOA) very easily. Under the hood, *thrust::make_zip_iterator* actually uses tuples of reference, that can be dereferenced for reading or writing operations. Zip iterators allows to extend all the previous concepts to a large range of applications.

3.2.4 GPU in iterative tomographic reconstruction

3.2.4.1 The matrix-vector product

As seen in the section 3.3.1, most of the iterative renconstruction technics are based on a set of linear equation. However, in typical real experiments settings in CBCT, where one image pixel account for one measurment, the number of measurments can vary from 5.10^7 (200 projections of size 500×500 pixels) to 5.10^8 (360 projections of 1200×1200 pixels). In the mean time, a classical reconstruction grid can count from $256^3 \approx 2.10^7$ to $1536^3 \approx 4.10^9$ volume elements. Representing the tomographic model, which takes into account every voxel contribution to every single acquisition pixel would require a matrix of size 10^{15} to 10^{18}, the storage of such matrix in single precision floating point format would require up to 10 exabytes, which is completely unfeasible in practice.

Fortunately, in most of the models, voxels in projective geometry have a relatively small footprint, overlapping a few pixels on every view, so that the resulting matrix has a high sparsity.

Sparsity of such matrices in parallel geometry has been studied in [**flores2014parallel**], and, for small reconstruction size, the author were able to use a on the shelf sparse solver. In [**iborra2016development**], the author did used an explicit matrix for the CBCT operators, and exploited it in order to assess condition number of the matrix, along various discretization parameters, and even used a QR based decomposition method to solve the problem. However, this work is based on a previous study: [**mora2008new**], where a specific geometry is assumed, in order to exploit numerous symmetries in the matrix.

In our case, we did performed a short study on the matrix sparsity, and concluded that an explicit matrix expression would not have been possible, even with the Siddon rendering method for an arbitrary geometry, see figure 3.4.

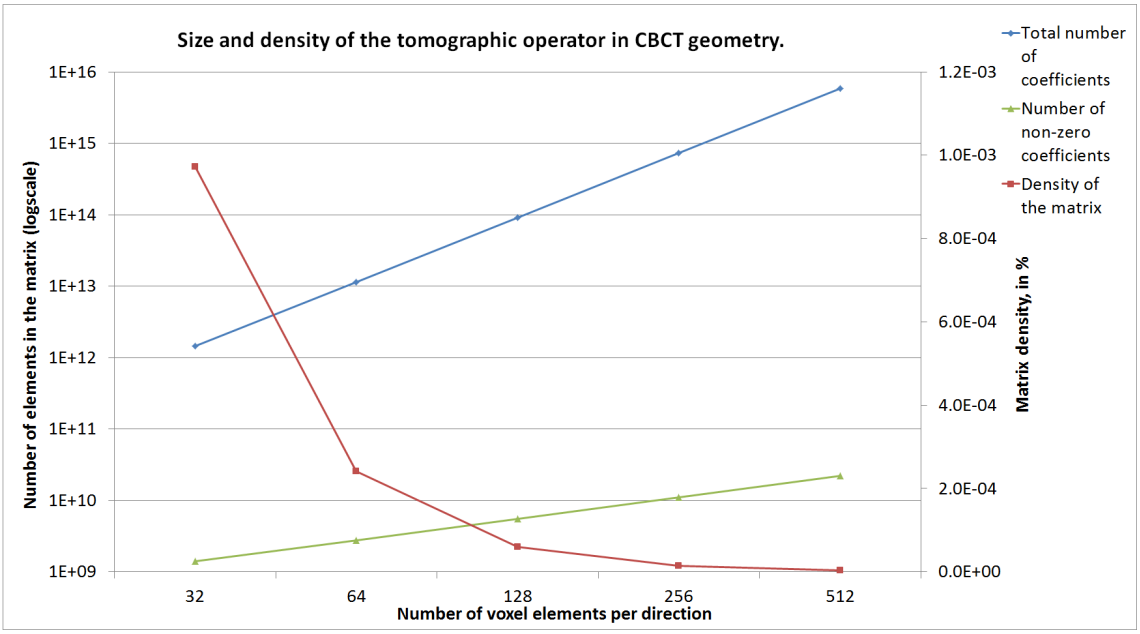


Figure 3.4: Matrix sparsity for the siddon projection model, using 40 views of 5.10^5 pixels.

3.2.4.2 GPU or CPU implementation

A large amount of litterature demonstrated that GPUs architecture was particularly well suited for accelerating tomographic operators. One of the most iconic publication being [**rohkohl2009technical**], where the authors provided, through a website, and an opensource benchmark platform, a way to compare the implementation of a voxel based backprojection operator in CBCT geometry. As of today (Nov. 2016), the best CPU implementation in the RabbitCT rankings is ranked 11th, and features only 3.56% of the performances of the best implementation.

However, it must be noticed that some advanced tomographic operators, cannot easily be implemented on GPUs, in the case they involve dynamic allocation for instance, see [**iborra2016development**], or researcher have been able to provide fast implementation by exploiting SIMD vectoization efficiently, see [**sampson:16:imt**] for instance.

We conducted a short study in order to assess the relevance of GPU versus CPU implementation of a simple matrix-projection model, using two compilers: gcc, and icc, and two parallelization libraries: OpenMP with its basic opensource runtime for gcc, and intel runtime for icc, as well as TBB[5].

In the next section, we will see the various possible way to define the matrix vector product Pv, seen in equation 3.2, and how they can be efficiently implemented on GPUs.

[5] *Thread Building Block*

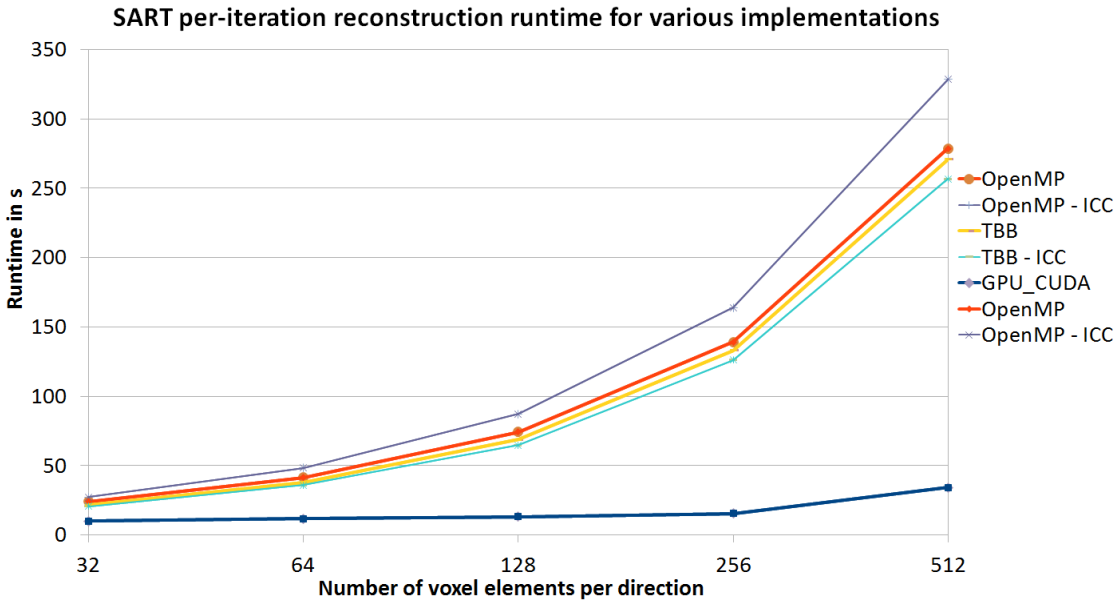


Figure 3.5: Per iteration runtime of the SART algorithm using Siddon projector, 40 views of 5.10^5 pixels. GPU:GTX680. CPU:Intel I7 3970X

3.2.4.3 Software architecture and data model for multi-GPU reconstruction

Although we have seen in section 3.2.4.2 that GPUs are particularly well suited to accelerate tomogaphic operators because of their ability to execute code in parallel, one may be interested in parallelizing tomographic operator execution on multiple GPUs. Designing an efficient software architecture that can parallelize various types of tomographics operators implementation over multiple GPUs is a challenging task, the authors of the book in [**jia2015graphics**] proposed an interesting set of tools to guide the design of such software.

Here, we will focus on the memory model we chose in order implement multiGPU reconstruction. Our aim was to design a robust software architecture, as much as possible compliant with the open-close principle, loosely coupled, that should allow us to perform any combination of GPU based tomographic operator for forward and back projection. In addition, one of the constraint was that the software should not rely on strong apriori on the number of accelerators, PCIe network configuration, homogeneity of the hardware, nor on the available memory on each of the GPUs, precluding from using peer-to-peer communication for instance. The use of unified virtual addressing, and later unified memory, that was introduced in cuda 6, was also impractical in our case because of the fact that managed allocations needed as much memory on the host, as on the device.

The enhancement of "cudaManagedMemory", towards a completely transparent memory management system, using on the fly page migration, etc... is very interesting. Unfortunately, this features came with the most recent release of cuda 8, and its support is very limited for hardware older than the current generation. In addition unified memory does not free the

developer from managing the specific 2D/3D memory layout that make tomographic operators very efficients, like texture memory.

In this framework, we designed a homemade multi-GPU memory management system, whose purpose was to allow for streaming both volume and projection data into each GPU, for further processing.

We figured the basic idea of this memory management system on the figure 3.6.

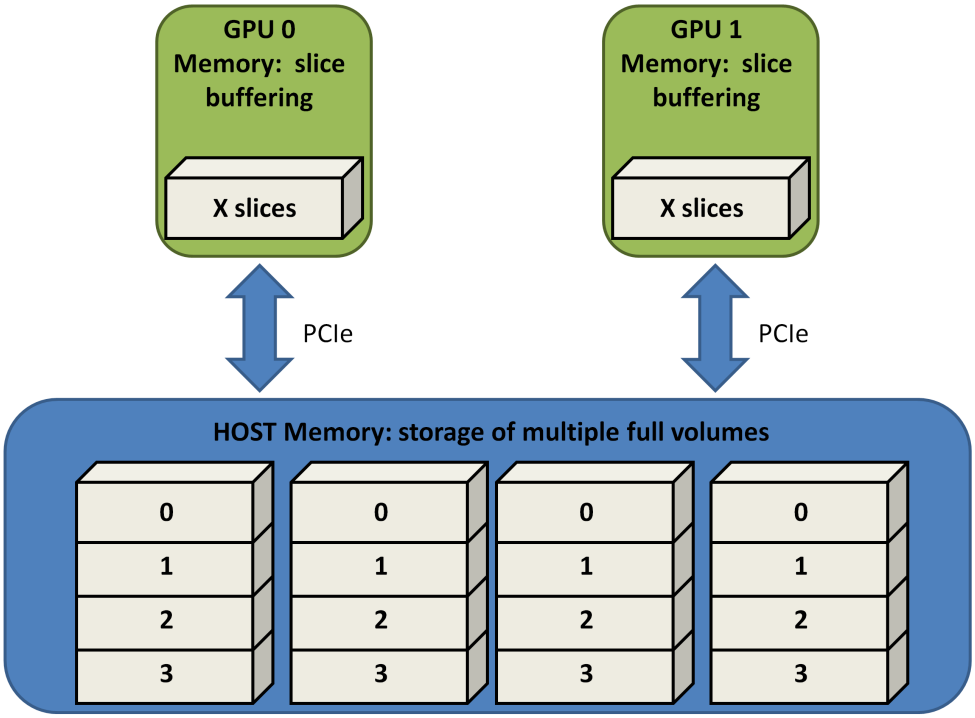


Figure 3.6: Simplified memory model

To set-up this idea, we used a simple software pattern, presented on figure 3.7.

CudaTypedVector On the figure 3.7, we can see that the top class is *CudaTypedVector*, this class acts like an interface that defines all operations that a vector should implement: elementwise operations, as well as reduction operators, and binary operators (addition, multiplications, ...).

CudaStreamingCapableVector Right under the top class, we define the *CudaStreamingCapableVector* which is the one that implements at a high level, the streaming semantic, for wich we spotted 4 main patterns, that we will describe in section 3.2.4.4. Those 4 patterns have

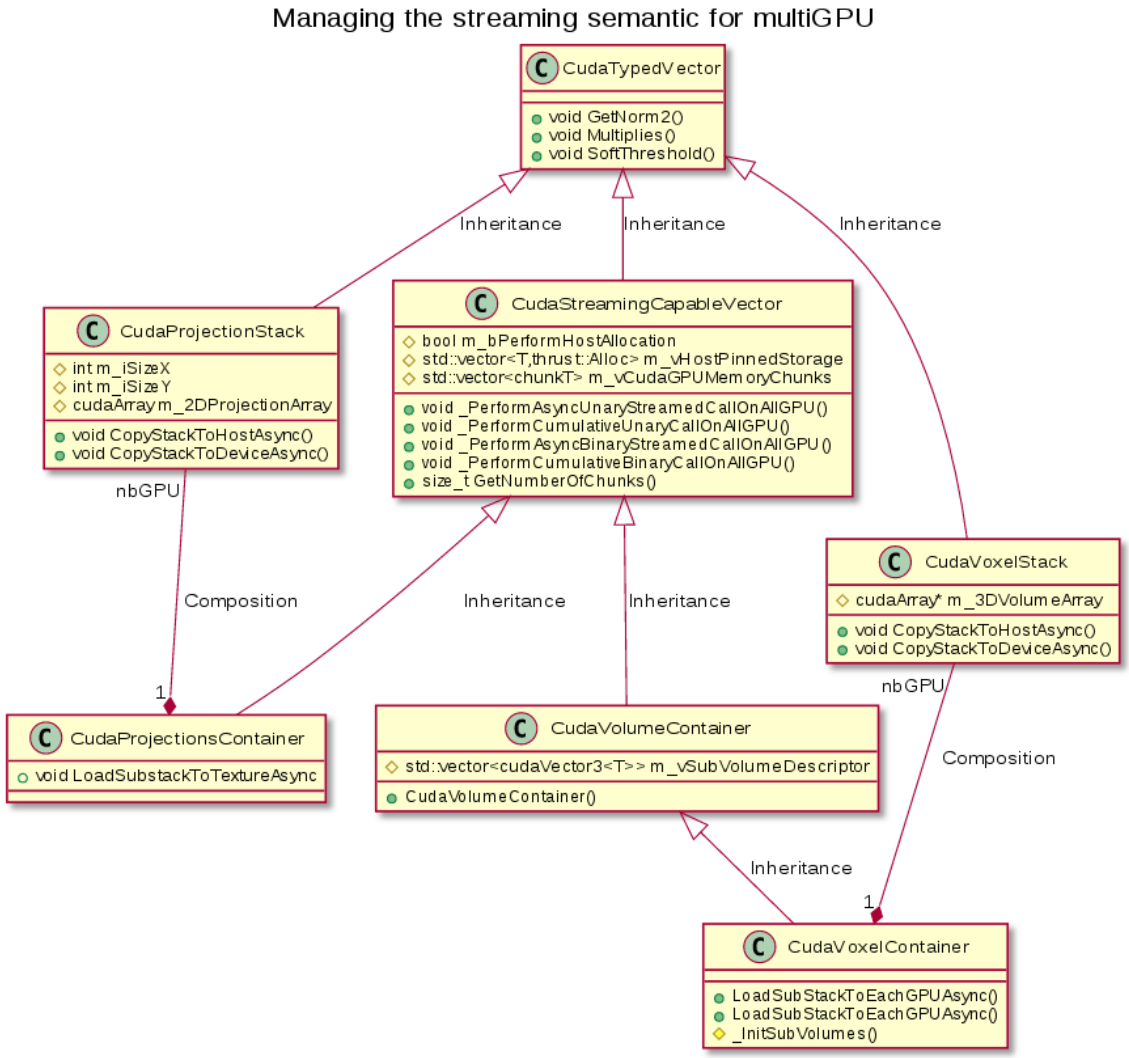


Figure 3.7: Simple software architecture to handle streaming semantic

been implemented in different methods, each defining a generic way to broadcast operators to independants chunks of memory, by launching asynchronously a preprocessing/computing/accumulating/postprocessing routine on all GPU recorded in the *m_vCudaGPUMemoryChunks* list. Although this does not figure on the UML diagram, this class is templated by the types of the memory chunks, so that it can be used in multiple context, for instance the daughter class can make access to the chunk vector and use the contained objects with their full capabilities.

One of the most important responsibility of *CudaStreamingCapableVector* is also to keep a reference on the main host memory buffer. In our case, this later is stored as a *std::vector* using a *thrust::pinned_allocator* to ensure maximal performance, and asynchronous behaviour for host-device related memory copy. If there is no GPU, or if some operations can be executed faster on the host, the class can decide to use its parent implementation on the full host vector range, otherwise, it uses one of the 4 parallelization method.

CudaVolumeContainer* and *CudaProjectionsContainer These two classes inherit from *CudaStreamingCapableVector*. They are used to handle the streaming semantic properly, based on the dimensionality of the data, and the access method. One should notice that the *CudaVolumeContainer* manage 3D volumes, that may need to be accessed using read-only trilinear filtering texture access. *CudaProjectionsContainer* also has some specific capabilities, as it manages stack of 2D projections. These objects are responsible for constructing the N memory chunks that will be stored in their parent list: *m_vCudaGPUMemoryChunks*, where N stands for the number of cuda-compatible GPUs detected by the system.

CudaProjectionsStack CudaVoxelStack Those two classes stands for streaming buffers on individual GPUs: either regular chunks of memory, or layout-specific arrays, with specific textures API handles.

3.2.4.4 GPU for vector-vector operations

GPU implementation of simple linear algebra operators One question that may arise while designing a GPU or multi-GPU based reconstruction application is wether or not it is interesting to perform vector-vector operations on the GPU. To answer this question, one has to define and characterize what are vector-vector operations. In our case, we derived 4 different patterns that covers most of the classical linear algebra operators on vectors:

Pattern	Unary transformation	Binary transformation	Reduction	Binary Reduction
Examples	scalar multiplication, soft thresholding, filling with sequence or random values, any unary operator...	addition, substraction, element wise multiplication, saxpy, any binary operator...	$L_0, L_1, L_2 \ldots$ L_∞ norm	Dot Product

In order to allow for a multi-GPU parallelization of all these operations, we designed simple code skeleton for each of these patterns, implemented in the *CudaStreamingCapableVector* class as:

- *_PerformAsyncUnaryStreamedCallOnAllGPU()* : manages Unary transformation
- *_PerformCumulativeUnaryCallOnAllGPU()* : manages Reduction
- *_PerformAsyncBinaryStreamedCallOnAllGPU()* : manages Binary transformation
- *_PerformCumulativeBinaryCallOnAllGPU()* : manages Binary Reduction

All those method are in charge of copying chunks of input data to the GPU memory, launch the computation or reduction kernel, and then copy the result back to host if needed. The work-item list is defined during construction in the subclass, either *CudaProjectionsStack*

or *CudaVoxelStack*, and the actual scheduling of those work items over each GPU relies on OpenMP, such that heterogeneous systems can be addressed efficiently.

Experiments In order to assess the performance of our multiGPU parallelization model regarding a multithreaded CPU approach, we designed 4 benchmarks challenging each of the four pattern seen in the previous paragraph.

In all cases, the input data size was 1024^3 single precision floating point elements, the hardware platform featured PCIe Gen 3, 16× for each GPU, a part from the 3-GPU configuration, where the GTX 750 Ti was connected in PCIe Gen 2. The Core i7 3970X is a high-end CPU, featuring 6 core, with hyperthreading technology (12 threads).

In each experiment, we challenged the minimal size of the atomic data chunk to be sent to the GPUs, and reported the runtime. The CPU code was also based on the thrust library, but compiled with the OpenMP backend, instead of the Cuda Backend.

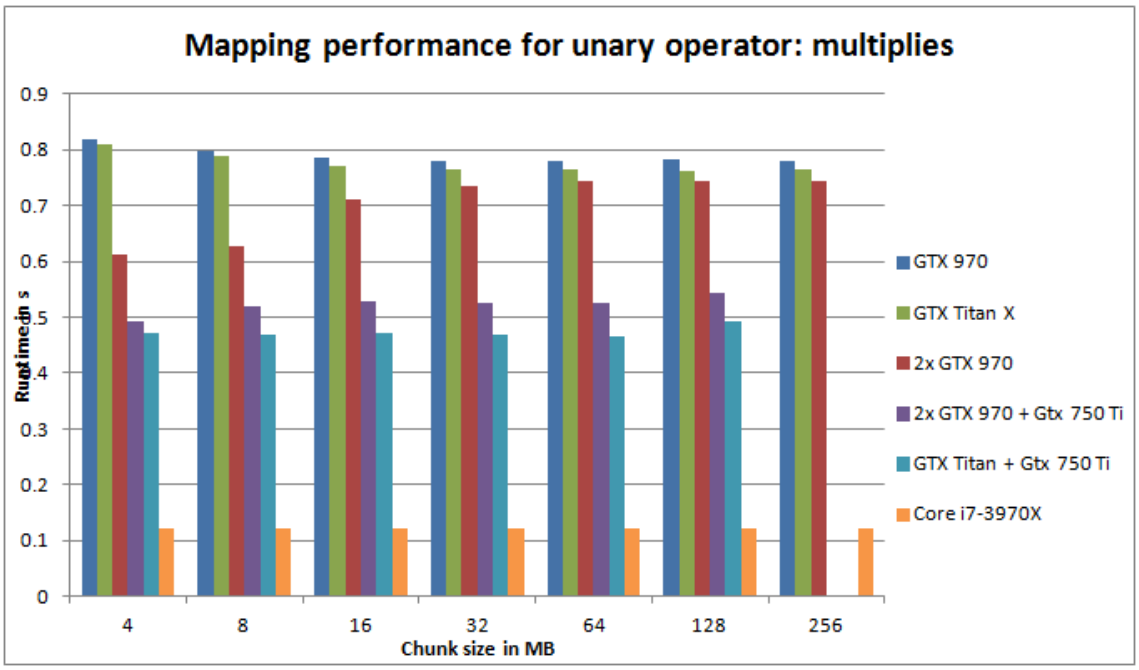


Figure 3.8: UnaryOperator

In all our experiments, it appeared that the GPU parallelization of the operators was less efficient than executing a parallel processing on the host through OpenMP. The main reason that explains those results is the PCIe bottleneck. On the unary operator, the host to device plus device to host copy duration was up to 50× longer than the processing time on the GPU. However, it can be seen on figure 3.10 that parallel reduction of a single vector on multiple GPUs was one of the most promising method. In this case, memory chunk size should be chosen carefully so that the overhead induced by API call, and the reduced performance of the GPU reduction over small data chunks would not be too penalizing. and the only one where our multi-GPU implementation was able to beat the CPU version.

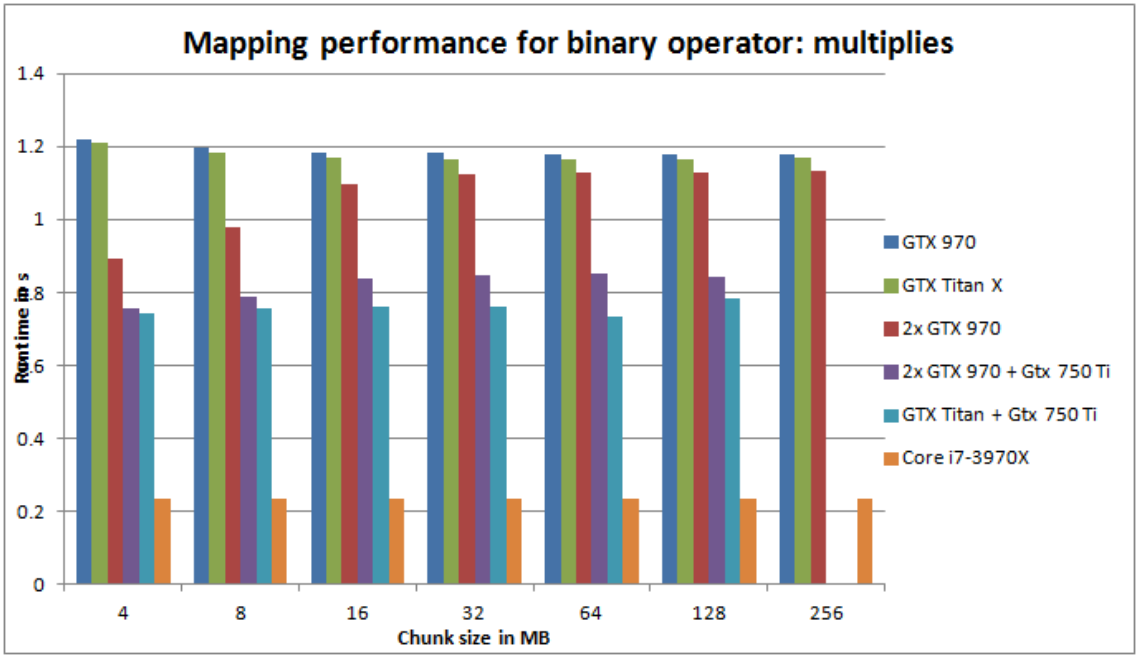


Figure 3.9: BinaryOperator

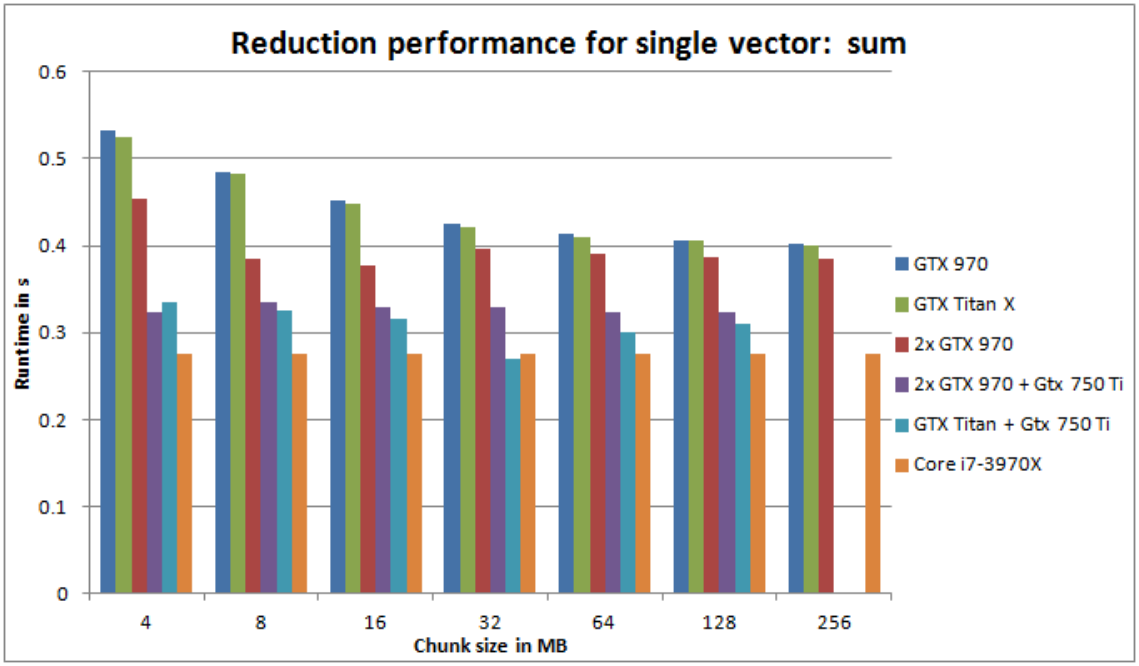


Figure 3.10: UnaryReduction

Future work will include a redesign of the individual GPU parallelization code to feature more fine-grained workload, able to overlap each other copy and computations process.

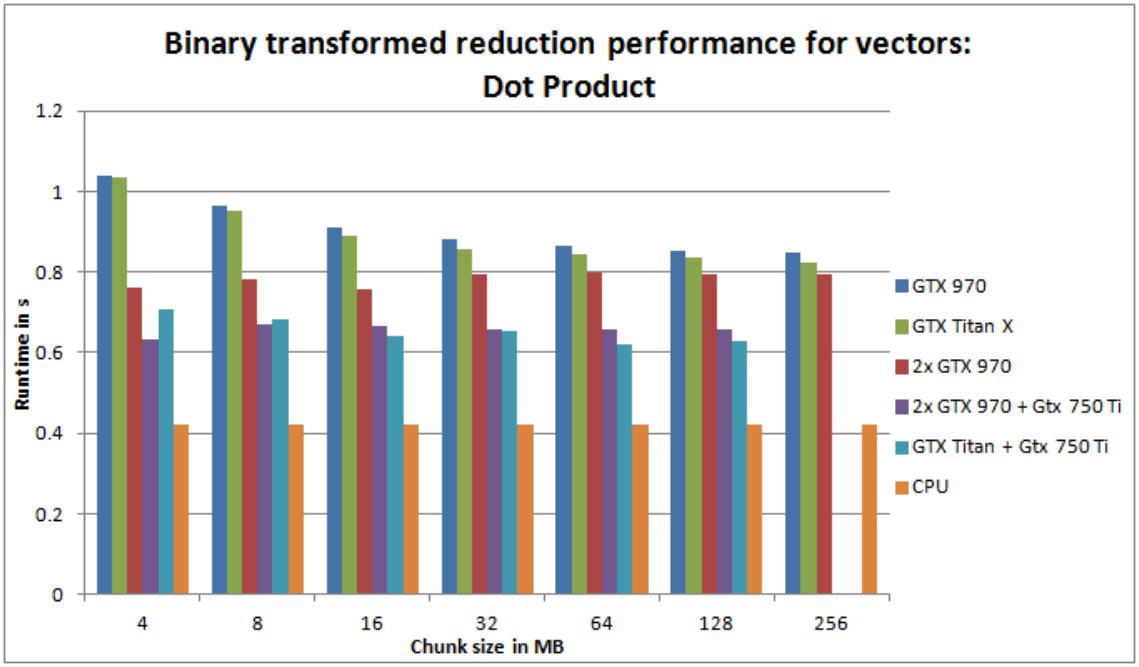


Figure 3.11: BinaryReduction

3.3 Imaging models for CBCT tomography

3.3.1 Algebraic formulation

The idea of algebraic reconstruction method, is to discretize the tomographic problem using the tools seen previously, it means that we will model the integral relation exposed in section 2.1.4 as a weighted sum of voxels attenuation coefficient times the geometrical contribution of this voxel to the set of X-Ray beams hitting the detector bin under consideration.

The projection equation now reads

$$\sum_{j=0}^{N_{vox}-1} \mu[j] c_{ij} = -log\left(\frac{p_i}{p0_i}\right) \tag{3.1}$$

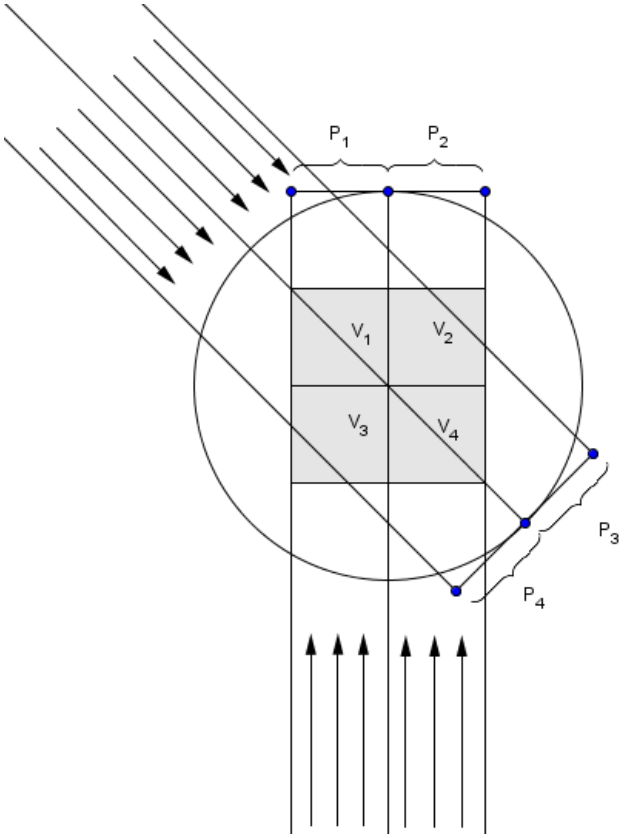
Where

- N_{vox} is the total number of grid nodes, or equivalently, the number of voxels
- $\mu[j]$ is the attenuation value of the j^{th} voxel
- c_{ij} is the geometrical contribution of the j^{th} voxel, to the attenuation measured on the i^{th} detector bin

If we arrange all N volume elements into a vector $\vec{V}$, and do the same for all K detector bin attenuation values so that we have another vector $\vec{P}$, we can write the tomographic problem in a matrix form:

$$\vec{P} = R\vec{V} \tag{3.2}$$

Where R is a $K \times N$ matrix, called the projection operator, we figured a simple schematic in order to illustrate this model, see figure 3.12



$$P_1 = r_{11}V_1 + r_{12}V_2 + r_{13}V_3 + r_{14}V_4$$
$$P_2 = r_{21}V_1 + r_{22}V_2 + r_{23}V_3 + r_{24}V_4$$
$$P_3 = r_{31}V_1 + r_{32}V_2 + r_{33}V_3 + r_{34}V_4$$
$$P_4 = r_{41}V_1 + r_{42}V_2 + r_{43}V_3 + r_{44}V_4$$

$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix} = \begin{bmatrix} r_{11} & \dots & r_{14} \\ \vdots & \ddots & \vdots \\ r_{41} & \dots & r_{44} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{bmatrix}$$

$$\vec{P} = R\vec{V}$$

Figure 3.12: Algebraic formulation of the tomographic reconstruction problem

3.3.2 Geometric consideration on the CBCT geometry

Now that the scope of our study regarding the modelization of the problem has been defined, we will give a short insight about the specific topic of CBCT system geometry. A good introduction on the topic of CBCT geometry can be found in [**galigekere2003cone**], in this section, we will show how the basics of projective geometry can be used to define tomographic operators.

First, we must recall, that any CBCT imaging system, with a flat panel detector, with no specific source or electro-magnetic field induced distortion can be modeled using the pinhole

camera model. Using this well known model, the X-Ray source can be identified as the pinhole, and the flat panel detector as the image plane. The 3D object is in general located between the source, and the image plane.

The geometrical projection operation in this case, amounts to apply an affine transformation, from a 3D coordinates point in a fixed world coordinate system, to a 2D projection coordinates, standing for the coordinates of the pixels.

3.3.3 CBCT geometry and projection matrices

For the sake of simplicity, a simple trick can be used to change the affine transformation defined above into a linear transform by adding a virtual coordinate to the 3D world coordinate system, that will always equal to 1 : $\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$.

In a second time, as we will use the framework of projective geometry, the output coordinates of a projection point will be expressed in homogeneous coordinates, hence any projection point onto the flat panel detector will be defined using 3 coordinates, as $\begin{pmatrix} u \\ v \\ w \end{pmatrix}$ instead of only 2.

In our context, those 3 coordinates will actually define a ray, that passes through the X-Ray source, and also throug the flat panel, where the last coordinate will be respectively equal to $w = 0$, and $w = 1$. For all points defined in projective geometry using this system, the 2D projections coordinates over the flat panel detectors pixels will be obtained as follows:

$$\begin{pmatrix} u_{pix} \\ v_{pix} \end{pmatrix} = \begin{pmatrix} \frac{u}{w} \\ \frac{v}{w} \end{pmatrix} \tag{3.3}$$

Now that we have defined the framework that will be used to handle geometric modeling of our system, we can see that the actual transformation from the 3+1D to the 2+1D coordinate system, can be written as a 3×4 matrix P

3.3.3.1 Projection matrices

Let P be our projection matrix, as defined in the introduction of section 3.3.3, in this case, the projection of a point $x_{\vec{x,y,z}}$ from the 3D world, onto a the flat panel detector, at pixel $p_{u,v}$ can be expressed as

$$P \cdot \vec{x_{x,y,z}} = p_{u,v} \tag{3.4}$$

$$\begin{pmatrix} P(0) & P(1) & P(2) & P(3) \\ P(4) & P(5) & P(6) & P(7) \\ P(8) & P(9) & P(10) & P(11) \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{3.5}$$

as the vector $\begin{pmatrix} P(3) \\ P(7) \\ P(11) \end{pmatrix}$ stands for a position invariant translation in the projective space we will call it T, we may actually be interested in dropping it later, to switch from a linear back to an affine transformation. To do so, we define the 3×3 matrix P_{sub} :

$$P_{sub} = \begin{pmatrix} P(0) & P(1) & P(2) \\ P(4) & P(5) & P(6) \\ P(8) & P(9) & P(10) \end{pmatrix} \tag{3.6}$$

The matrix P, can be seen as the product of 4 matrices $P = DIEG$, each defining a specific physical transformation, successively in 5 coordinate systems or space C_X where:

- $G : C_{fixed\ \ grid\ \ 3D} \rightarrow C_{fixed\ \ orth\ \ 3D}$ is 4×4 equivalent to a 3D grid to grid transformation matrix.
- $E : C_{fixed\ \ orth\ \ 3D} \rightarrow C_{rotated\ \ orth\ \ 3D}$ stands for a 3×4 extrinsic matrix.
- $I : C_{rotated\ \ orth\ \ 3D} \rightarrow C_{orth\ \ 2D}$ stands for a 3×3 intrinsic matrix.
- $D : C_{orth_2D} \rightarrow C_{det\ \ grid\ \ 2D}$ is also a 3×3 matrix, equivalent to a 2D grid to grid transformation matrix

G matrix It allows to transform a 4-uplet of coordinates from an arbitrary regular grid coordinate system in 3D into its 4-uplet equivalent in the proper orthogonal coordinate system with the same origin:

$$G = \begin{pmatrix} D_{3D\ \ grid} & 0 \\ 0 & 1 \end{pmatrix} \tag{3.7}$$

This 4×4 matrix contains the non-singular 3D grid matrix $D_{3D\ \ grid}$, which is made of 3 vectors, where each column defines one of the periodization vector of the regular grid that has been chosen to discretize data, as defined in section 2.2.4.

E matrix The extrinsic matrix E can be thought of as the concatenation of a 3×3 unitary matrix representing a rotation in $\mathbb{R}^3$, and a vector representing a translation:

$$E = \left(\begin{pmatrix} r_0 & r_1 & r_2 \\ r_3 & r_4 & r_5 \\ r_6 & r_7 & r_8 \end{pmatrix} \begin{pmatrix} T_x \\ T_y \\ T_z \end{pmatrix} \right) \tag{3.8}$$

$$= \begin{pmatrix} R & \vec{T_{xyz}} \end{pmatrix} \tag{3.9}$$

R is a rotation matrix from the $SO(3)$ group, and as such it can be factorized as a product of 3 simple rotation matrices, of angles ψ, θ, ϕ along each axis of the coordinate system. It allows to change from an arbitrary fixed cartesian world coordinates system $C_{fixed\ orth\ 3D}$, to a detector axis aligned coordinate system $C_{rotated\ orth\ 3D\ centered}$. In this intermediate coordinate system, the x and y axis are colinear, respectively with the $\vec{u}$ and $\vec{v}$ vectors of the orthonormal coordinate system of the detector plan: $C_{orth\ 2D}$. In this framework, z axis is normal to the surface of the FPD. Before the application of the translation $\vec{T_{xyz}}$, the origin of both coordinate systems coincides in one point called the isocenter.

The vector $-\vec{T_{xyz}}$ corresponds to the coordinates of the source in $C_{rotated\ orth\ 3D\ centered}$, such that the coordinate system $C_{rotated\ orth\ 3D}$ origin coincides with the X-Ray source.

I matrix The intrisic matrix I is a 3×3 triangular matrix, with the following form:

$$I = \begin{pmatrix} 1 & 0 & u_0 \\ 0 & 1 & v_0 \\ 0 & 0 & \frac{1}{f} \end{pmatrix} \tag{3.10}$$

Where we can give the following interpretation:

- u_0 and v_0 stands for the coordinates of the projection of the source onto the projection plane.
- f stands for the source-detector distance, in the 3D fixed world coordinate system.

D matrix The matrix D presented earlier accounts for the detector pixel grid, it allows a linear transformation, from a 2D orthonormal coordinate system, to a pixel based detector grid coordinate system.

$$D = \begin{pmatrix} D_{2d\ grid}^{-1} & 0 \\ 0 & 1 \end{pmatrix} \tag{3.11}$$

Where $D_{2d\ \ grid}$ is the non-singular 2×2 matrix, that contains the pixel grid periodization column vector, following the regular grid definition we provided in 2.2.4. This definition allows for any regularly sampled version of the 2D data coming from the detector.

We figured those coordinate systems on the figure 3.13.

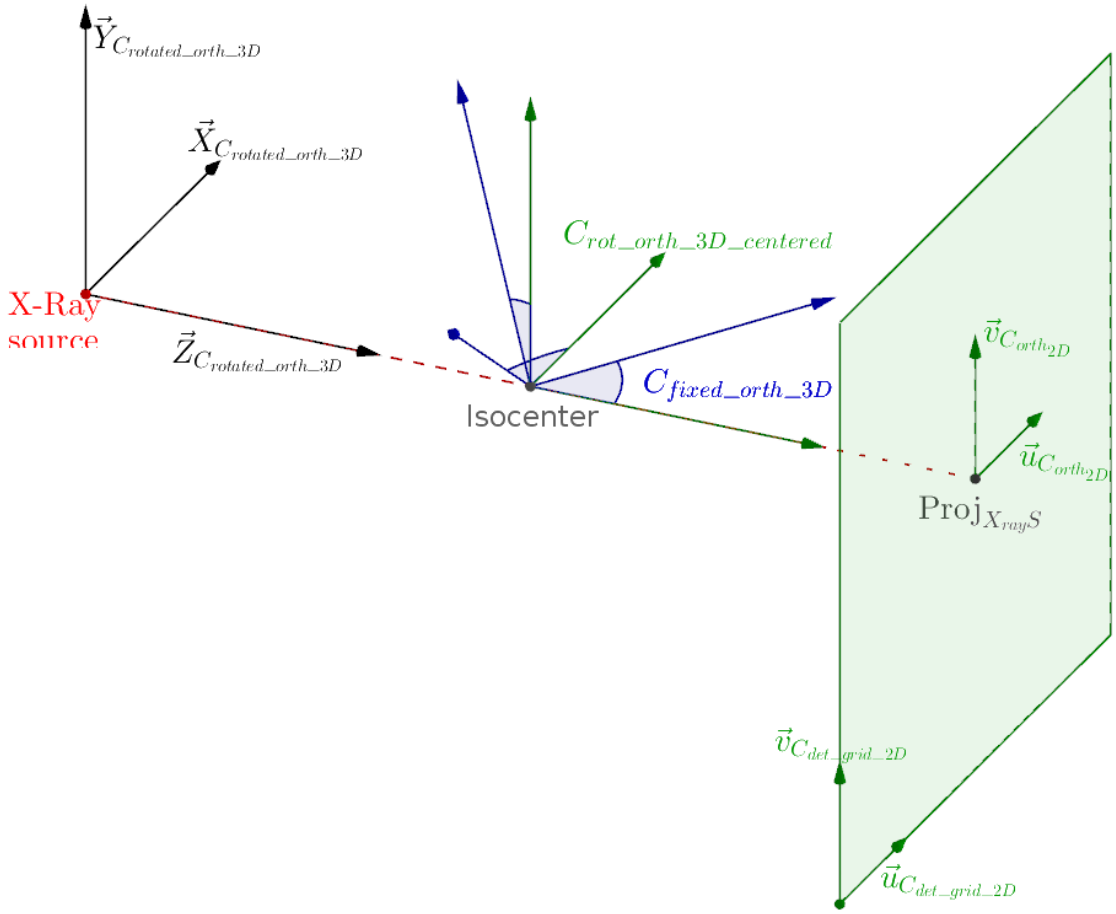


Figure 3.13: Simplified schematic of a CBCT system geometry

3.3.3.2 Calibration of projection matrices

Although the calibration of a tomographic system is a topic that goes beyond the scope of this thesis, we must highlight the fact that this aspect is of crucial importance to allow for a proper reconstruction. Most of the algebraic reconstruction models can handle various inconsistency in the data, related to photon noise, detector noise, physical apriori on the X-Ray spectrum, photon scattering, etc ... But assume a perfect modelization of the system geometry, obtained through an anterior calibration process, see for instance [**rougee1993geometrical**], [**cho2005accurate**].

Recent advances in integral geometry and computer vision allowed to estimate and cor-

rect for geometrical inconsistencies in CBCT systems, based on ECC[6] for instance, see [**aichert2015epipolar**].

In the general case, most of the calibration procedures use a physical phantom containing radiopaque elements, generally beads, of known coordinates in the coordinate system of the object, sometimes called fiducial points. Each of the elements should be designed such that detection and estimation of the projection coordinate of their center can be easily performed on the projection image. The pattern should be such that there is no ambiguity in pairing 2D points coordinates with each of the 3D point.

Using a calibration phantom, and an adapted detection, and identification algorithm, we should be given for each view a set of n pairs of coordinates:

- the 3+1D coordinates of a pattern element in the arbitrary object based coordinate system, expressed as:

$$B_n = \begin{pmatrix} x_0 & x_1 & \dots & x_{n-1} \\ y_0 & y_1 & \dots & y_{n-1} \\ z_0 & z_1 & \dots & z_{n-1} \\ 1 & 1 & \dots & 1 \end{pmatrix} \tag{3.12}$$

- the 2+1D coordinates of the previous element projection, expressed in terms of detector pixel coordinates, that were arbitrarily chosen so that their last coordinate is normalized.

$$C_n = \begin{pmatrix} u_0 & u_1 & \dots & u_{n-1} \\ v_0 & v_1 & \dots & v_{n-1} \\ 1 & 1 & \dots & 1 \end{pmatrix} \tag{3.13}$$

Next, there are least 2 optimization problems that can be casted in order to recover the system geometry:

Implicit calibration The implicit linear formulation: simply amounts to find the P matrix that make the B_n and C_n to be consistent with each other such that we have:

$$PB_n = C_n \tag{3.14}$$
$$B_n^\intercal P^\intercal = C_n^\intercal \tag{3.15}$$

For the sake of simplicity, the matrix $P^\intercal$ can be linearized as a vector P_{lin}:

$$P_{lin} = \begin{pmatrix} P(0) \\ P(1) \\ \vdots \\ P(11) \end{pmatrix} \tag{3.16}$$

[6] *Epipolar Constistency Conditions*

And the B_n and C_n matrices, resized accordingly:

$$B_{nlin} = \begin{pmatrix} \begin{pmatrix} x_0 & y_0 & z_0 & 1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & x_0 & \cdots & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & x_0 & \cdots & 1 \end{pmatrix} \\ \begin{pmatrix} x_1 & y_1 & z_1 & 1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & x_1 & \cdots & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & x_1 & \cdots & 1 \end{pmatrix} \vdots \\ \begin{pmatrix} x_{n-1} & y_{n-1} & z_{n-1} & 1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & x_{n-1} & \cdots & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & x_{n-1} & \cdots & 1 \end{pmatrix} \end{pmatrix} \tag{3.17}$$

$$C_{nlin} = \begin{pmatrix} u_0 \\ v_0 \\ 1 \\ u_1 \\ v_1 \\ 1 \\ \vdots \\ u_{n-1} \\ v_{n-1} \\ 1 \end{pmatrix} \tag{3.18}$$

In this case, one has to solve the linear set of equation $B_{nlin}P_{lin} = C_{nlin}$. Alternatively, a more realistic optimization problem is the least square version of the previous equality: $\underset{P}{min}\|B_{nlin}P_{lin} = C_{nlin}\|_2^2$ where we are looking for the minimal reprojection error in the least square sense.

Explicit calibration The previous method is pretty easy to implement, unfortunately, it is not truly consistent with the physical reality of the geometry model. Indeed, it may looks like there are 12 DOF[7] in our system geometry, however it is not the case, looking at the description of our geometry model in 3.3.3.1, we see that we have actually 3 angles ψ, θ, ϕ, and 3 coordinates T_x, T_y, T_z to retrieve in the E matrix, and 3 distances u_0, v_0, f to retrieve in the I matrix, which makes a total of 9 DOF.

Taking all the geometrical relationships described in the section 3.3.3.1, we can setup a non-linear calibration optimization problem, that can be casted using a least square approach too, this is called explicit calibration. However this problem is far more complicated to solve, and, as we will show in the next sections, we will be able to define tomographic operators without having access to the explicit geometric system parameters.

[7] *Degree Of Freedom*

3.3.3.3 Factorization of projection matrices

From what we have seen in section 3.3.3.2, performing an implicit geometric calibration with a dedicated phantom may not be an extremely challenging task, assuming that fiducial point extraction and identification are given. However, performing the full matrix factorization of the form $DIEG$ seen in section 3.3.3.1 may not be as trivial.

In practice, the D^{-1} and G^{-1} matrices are known, so that the problem amounts to solve the IE factorization. The fact that we can find the source coordinates using a simple method exposed in 3.3.3.4, allows us to focus on factorizing IR, the product between I and the rotation matrix R. The fact that one of this matrix is from $SO(3)$, and the other is triangular superior can be leveraged by using a generic QR factorization method, on $(IR)^{-1}$ (in order to obtain an equivalent RQ decomposition. This approach has been used for instance in [**fusiello2000compact**]. However in practice, it appeared that small inconsistencies in the projection matrices obtained from the calibration process yielded RQ factorization where the intrisic matrix had a nonzero detector skewness.

Instead of using a re-orthogonalization process, we decided to focus in this chapter on tomographic projectors that were only based on projection matrices, ignoring any other geometric informations.

3.3.3.4 Finding the source coordinates

For any point x in 3D, that projects onto the point p when multiplied by the matrix P, it is easy to see that any displacement of x in the direction of the source s will not change the projection coordinates. We also recall that, if the homogeneous coordinates of a point p are multiplied by a non-zero coefficient $k \in \mathbb{R}^*$, then the resulting coordinates represents the same point. We can summarize the above property as:

$$\forall \lambda, x \in \mathbb{R} \times \mathbb{R}^3, \exists \lambda_2 \in \mathbb{R}^* \quad \text{s.t} \quad P(x + \lambda(s - x)) = \lambda_2 Px \tag{3.19}$$

$$(1 - \lambda)Px + \lambda Ps = \lambda_2 Px \tag{3.20}$$

$$(1 - \lambda - \lambda_2)Px + \lambda Ps = 0 \tag{3.21}$$

$$(1 - \lambda - \lambda_2)Px = -\lambda Ps \tag{3.22}$$

There are multiple cases:

- $\lambda = 0$ and $x = \vec{0}$, in this case, the property holds for any value of s and λ_2
- $\lambda = 0$ and $x \neq \vec{0}$, in this case, there are two other option to consider:

  $x \in ker(P)$: in this case, the property holds for any value of s and λ_2

  $x \notin ker(P)$: in this case, the property holds for any value of s if $\lambda_2 = 1 - \lambda$

- $\lambda \neq 0$ and $x = \vec{0}$, in this case, the property holds only if $s \in ker(P)$

- $\lambda \neq 0$ and $x \neq \vec{0}$, in this case, there are two other option to consider:

  $x \in ker(P)$: in this case, the property holds only if $s \in ker(P)$

  $x \notin ker(P)$: in this case, the property holds only if $s \in ker(P)$ and $\lambda_2 = 1 - \lambda$

We can conclude that the above definition for the source coordinates s holds only when $s \in ker(P)$. For the sake of simplicity, we will look for the subspace of the null space of matrix P, that intersect the hyperplan $x^\intercal \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = 1$. This will give us the coordinates of the source in the form $\begin{pmatrix} s_x \\ s_y \\ s_z \\ 1 \end{pmatrix}$

We also define the matrix M as $M = P_{sub}^{-1}$, that we will be using in the next sections:

$$Px = 0 \tag{3.23}$$

$$\left(\begin{pmatrix} P(0) & P(1) & P(2) \\ P(4) & P(5) & P(6) \\ P(8) & P(9) & P(10) \end{pmatrix} \begin{pmatrix} P(3) \\ P(7) \\ P(11) \end{pmatrix} \right) \cdot \begin{pmatrix} s_x \\ s_y \\ s_z \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{3.24}$$

$$\begin{pmatrix} P(0) & P(1) & P(2) \\ P(4) & P(5) & P(6) \\ P(8) & P(9) & P(10) \end{pmatrix} \cdot \begin{pmatrix} s_x \\ s_y \\ s_z \end{pmatrix} + \begin{pmatrix} P(3) \\ P(7) \\ P(11) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{3.25}$$

$$P_{sub} s_{x,y,z} + \vec{T} = 0 \tag{3.26}$$

$$s_{xyz} = P_{sub}^{-1} \cdot -\vec{T} \tag{3.27}$$

$$s_{xyz} = M \cdot -\vec{T} \tag{3.28}$$

$$\tag{3.29}$$

In the following developments, we assume that the 3D coordinates of the source, named s_x , s_y , s_z, are known, as well as the matrix M.

3.3.3.5 Defining a ray in CBCT geometry

Using the same strategy presented in section 3.3.3.4, we will now try to find the unitary direction vector $\vec{d}$ of the form $\vec{d} = \begin{pmatrix} d_x \\ dy \\ d_z \\ 0 \end{pmatrix} = \begin{pmatrix} d_{xyz} \\ 0 \end{pmatrix}$, from $x(\vec{t}) = \vec{s} + t\vec{d}$ that describes the path between the source point s and the projection p of known coordinates $\begin{pmatrix} u \\ v \\ 1 \end{pmatrix}$:

$$\forall t \in \mathbb{R}^*, \exists \lambda \in \mathbb{R}^* \quad \text{s.t} \quad P(s+td) = \lambda p \tag{3.30}$$

$$tPd = \lambda p \tag{3.31}$$

$$P_{sub} d_{xyz} = \lambda_2 p \quad \text{with} \quad \lambda_2 = \frac{\lambda}{t} > 0 \tag{3.32}$$

$$d_{xyz} = \lambda_2 M p \tag{3.33}$$

We just need to set $\lambda_2 = \frac{1}{\|Mp\|}$ in order to ensure that $\vec{d_{xyz}}$ is a unitary vector. Lets write it down

$$\vec{d_{xyz}} = \frac{Mp}{\|Mp\|} \tag{3.34}$$

$$\vec{d_{xyz}} = \frac{1}{\|M \cdot \begin{pmatrix} u \\ v \\ 1 \end{pmatrix}\|} \begin{pmatrix} M(0)u + M(1)v + M(2) \\ M(3)u + M(4)v + M(5) \\ M(6)u + M(7)v + M(8) \end{pmatrix} \tag{3.35}$$

For simplification, we will express the normalization factor as K, so we can rewrite $\vec{d_{xyz}}$ as:

$$\vec{d_{xyz}} = \frac{1}{K} \cdot \begin{pmatrix} d_x \\ d_y \\ d_z \end{pmatrix} \tag{3.36}$$

$$= \frac{1}{K} \cdot \begin{pmatrix} M(0)u + M(1)v + M(2) \\ M(3)u + M(4)v + M(5) \\ M(6)u + M(7)v + M(8) \end{pmatrix} \tag{3.37}$$

3.4 Classical tomographic operators

3.4.1 Introduction

Modeling the ray imaging process is a topic that can be viewed from multiple perspectives. The interaction between matter and high energy electromagnetic wave such as X-Rays, at a microscopic level has been tackled by physicists, and will not be discussed here. In the framework of tomography, we will mostly be interested in designing fast and accurate linear models of X-Ray images rendering, and their adjoints.

3.4.2 Siddon projector

One of the first model of projection was designed by Siddon in [**siddon1985fast**]. This model assume an infinitely narrow beam, traveling from a source to the center of a detector bin, and intersecting the various polygons representing the volume elements. This methods allows for computing each source - detector bin trajectory independantly, hence it is called a ray-based approach. The actual contribution of each voxel to the current detector bin is calculated as the length of their geometrical intersection, as figured on the figure 3.14.

The strength of ray-based approach for projector, is that their implementation generally imply a redundant access to read-only memory for the volume, and independant write access to the projection memory accumulator. These advantages obviously becomes drawback for the backprojection scheme, where one should make independant and non redundant acces to read only projection memory, and redudant and concurrent writes to volume memory.

The ray based approach can be efficiently parallellized on both CPU and GPU hardware, and open-source software featuring this projector has been released, see for instance [**gao2012fast**].

From a signal processing point of view however, we can see that this model of infinitely narrow sampling beam will generate a non uniform sampling pattern across the volume in divergent geometry. In practice, for each view, the sampling frequency will be higher in the volume area closest to the source, and lower toward the detector. In practice this property can lead to aliasing artifact, if the "virtual", detector resolution is not well chosen, as studied in **??**.

3.4.3 Ray traversal with trilinear interpolation

Although some variation of Siddon's model have been proposed to optimize the rendering complexity, one of the most widely used projector was designed by Kohler, Turbell and Grass in [**kohler2000efficient**]. They proposed a model where the ray integration process was computed by traveling along the source-detector trajectory, and sampling the volume using a trilinear interpolation method, using step by step approach, as figured on figure 3.15.

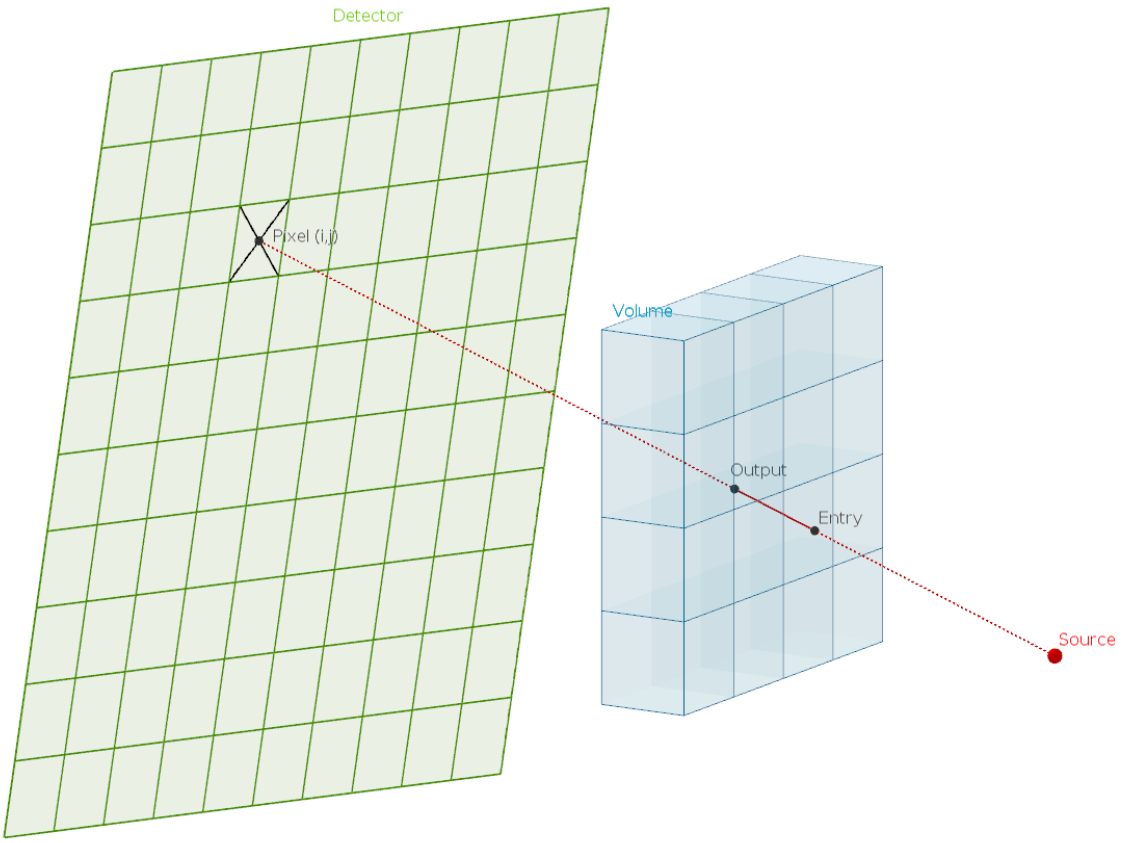


Figure 3.14: Siddon ray-based projector

The trilinear sampling operator $S(x, y, z)$, in a voxel based grid, centered at the corner of the $[0, 0, 0]$-indexed voxel reads as follows:

$$S(x, y, z) = \tag{3.38}$$
$$(1-\alpha)(1-\beta)(1-\gamma)T[i, j, k] + \alpha(1-\beta)(1-\gamma)T[i+1, j, k] + \tag{3.39}$$
$$(1-\alpha)\beta(1-\gamma)T[i, j+1, k] + \alpha\beta(1-\gamma)T[i+1, j+1, k] + \tag{3.40}$$
$$(1-\alpha)(1-\beta)\gamma T[i, j, k+1] + \alpha(1-\beta)\gamma T[i+1, j, k+1] + \tag{3.41}$$
$$(1-\alpha)\beta\gamma T[i, j+1, k+1] + \alpha\beta\gamma T[i+1, j+1, k+1] \tag{3.42}$$

Where:

- $i = floor(x_{dis})$, $\alpha = frac(x_{dis})$ and $x_{dis} = x - 0.5$
- $j = floor(y_{dis})$, $\beta = frac(y_{dis})$ and $y_{dis} = y - 0.5$
- $k = floor(z_{dis})$, $\gamma = frac(z_{dis})$ and $z_{dis} = z - 0.5$

This method had the advantage of greatly reducing the aliasing artifacts caused by the infinitely narrow beam model of Siddon, while keeping the ray-based approach.

Another advantage of this method, is that it maps very efficiently to GPU computing hardware, that generally features a hardware based trilinear filtering texture access, with a specific texture cache.

One of the drawback however, is that computing the adjoint of this tomographic operator, and in particular the adjoint of the trilinear sampling operator, called splatting, is far less friendly, which motivated the use of unmatched pair of projector and backprojector, see for instance [**zeng2000unmatched**] regading this type of design.

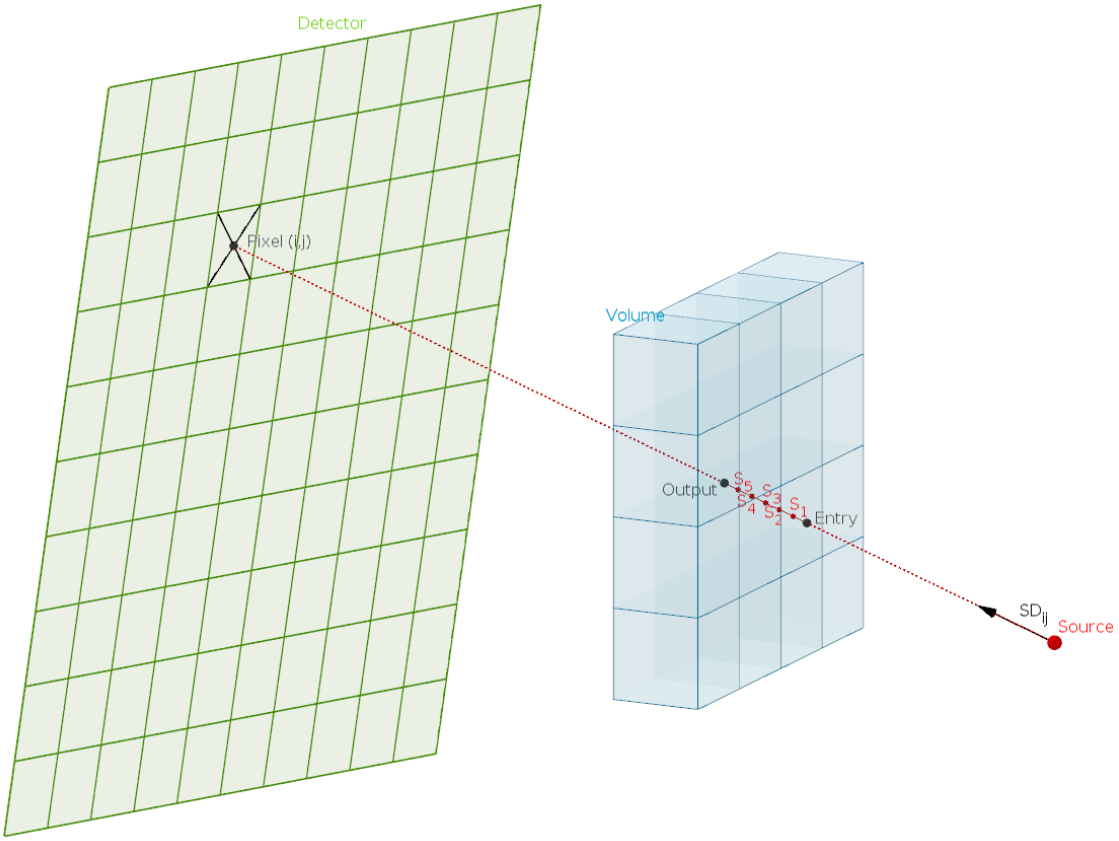


Figure 3.15: Ray casting based projector

3.4.4 Voxel based operator with interpolation

A completely different approach is to compute independantly the projection of every voxels of the volume onto the pixel detectors it contributes to. This approach is called voxel based projection. One advantage of this method is that, as the volume contains generally more voxels that there are pixels on each view, the problem is embarassingly parallel, and can be very easily mapped to parallel computing architectures.

Computing the projection of the center of a voxel onto the detector plane is easy, as seen in section 3.3.3.1. However, splatting the voxel value onto multiple neighbouring pixels can be computationally inefficient, as this operation generates multiple concurrent write to neighboring pixels.

The formula for the 2D splatting operator is as follows:

- $P[i,j] += (1-\alpha)(1-\beta)V_{xyz}$
- $P[i+1,j] += \alpha(1-\beta)V_{xyz}$
- $P[i,j+1] += (1-\alpha)\beta V_{xyz}$
- $P[i+1,j+1] += \alpha\beta V_{xyz}$

Where V_{xyz} is the voxel value to be splatted, and i, j are defined using the the coordinates $\begin{pmatrix} x_{proj} \\ y_{proj} \end{pmatrix}$ of the projection of the center of the current voxel:

- $i = floor(x_{dis})$, $\alpha = frac(x_{dis})$ and $x_{dis} = x_{proj} - 0.5$
- $j = floor(y_{dis})$, $\beta = frac(y_{dis})$ and $y_{dis} = y_{proj} - 0.5$

From a signal processing perspective, this approach has some inconvenience: in this model, it is assumed that a voxel, no matter its size, and no matter the projective geometry context, will always be projected onto 4 pixels. This model even disregards the resolution of the detector. This approach can lead to severe aliasing artifacts when the voxel footprint occupy far more than 2×2 pixels on the detector.

However, the corresponding backprojector can be very efficiently mapped to GPU hardware, as there are multiple redundant read-only memory access on the projection memory, and independant access to the volume memory.

3.4.5 Other approaches

Numerous strategy have been designed in order to overcome common limitations of tomographic projectors. One of the most successfull projector, which is used by many researchers is the distance driven projector, see [**de2004distance**], which is a voxel based approach that takes into account the voxel footprint, and propose a particularly elegant implementation. This approach was later extended with the separable footprint model, see [**long20103d**], that propose a very precise evaluation of a voxel footprint based on trapezoid. More recently, this approach has even been refined in [**ha:16:eab**], using a set of lookup tables. However, it is important to notice that these approaches make the assumption that the volume grid axis vectors are colinear with the detector grid axis, which is a strong assumption for people targeting arbitrary geometry.

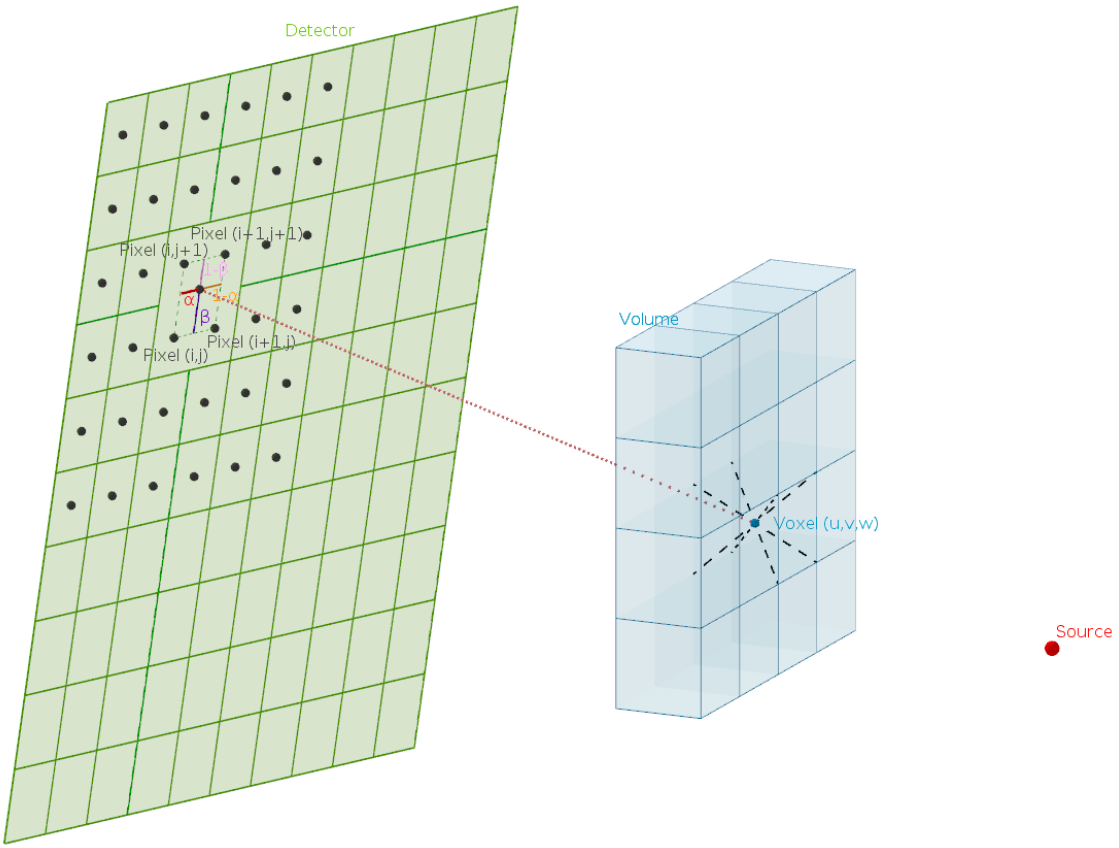


Figure 3.16: Voxel based projector with bilinear interpolation

Perfect evaluation of pyramid-voxel intersection in projective geometry, using the concept of LoR (line of response) has been derived in [**iborra2016development**]. However this approach imply dynamic memory allocation, and cannot be implemented in GPU for acceleration, plus it is also based on multiple non-trivial symmetry apriori explained in [**mora2008new**].

Among the voxel based methods that assume a radially symmetric volume element (blob), two advanced projectors were described in [**ziegler2006efficient**] and [**momey2013spline**]. The first method offers an extremely appealing tradeof between accuracy and computational cost, through the use of lookup tables. Unfortunately, the model is only valid for cylindrical detectors that have their main axis aligned with the volume grid axis. The second method was driven by a recent theory of efficient sampling, based on spline interpolation functions, and has been proposed by Fabien Momey: the spline driven projector. This method was able to provide a good approximation of the footprint of a spline RBF, with a separable function.

A completely different approach was derived for projecting objects based on an adaptive tetrahedralization of the scene. This mesh-based approach has been described in [**quinto2013tetrahedral**] and [**cazasnoves2015adapted**].

3.5 Blob based operator in CBCT geometry

In this section, we will develop a blob base projector, that will allow us to use more exotic volume discretization model, like the spherically symmetric elements seen in 2.2.7, in conjunction with the BCC[8] grid seen in 2.2.4.

3.5.1 From sphere projection to Conic equations

A generic framework for projecting an ellipsoid from a 3D world onto a 2D plane can be found in [**eberly1999perspective**]. According to this work, it is clear that, the cone beam projection shadow of a sphere, such as a truncated spherically symmetric volume element is a conic, and that it should be easy to derive the equation of its intersection with a plane, as a simple conic section.

In this section, we will be using the framework of ellipsoid, conic section and projective geometry to derive the footprint of our truncated blob model on the X-Ray flat panel projection plane.
To do so, we recall that we derived the definition of a ray $x(\vec{t})$ in section 3.3.3.5 , which has the following expression:

$$x(\vec{t}) = \vec{s} + t\vec{d} \tag{3.43}$$

- $\vec{s}$ being the coordinates of the source point
- $\vec{d}$ being a unit length vector standing for the direction
- $t > 0$ being the distance between the source and the tip of the ray

The projection of a quadric defined in 3D, onto a 2D plane can be simply defined by adding a set of constraints to the ellipsoid equation. The set of constraint simply restricts the ellipsoid to the set of points that also belong to a plan of equation $n^\intercal x = \lambda$.

In the next part, we will show why how to set up those two equations for any arbitrary cone beam geometry, in order to project our spherically symmetric volume elements.

3.5.2 Computing sphere projection from arbitrary projection matrices

3.5.2.1 Ray-sphere intersection

Given a projection matrix P, we have seen in section 3.3.3.5 how to define a ray $x(t)$ that intersect the detector plane at coordinate p, and in particular, we derived its expression in

[8] *Body Centered Cubic Lattice*

equation 3.3.3.5, which will be used in this section.

In order to project a truncated radial blob function, we will be first interested in defining the set of all rays touching the surface of the sphere of center c and radius r, which is one of the most simple ellipsoid.

$$\exists t \in \mathbb{R}_{+\star} \text{ s.t } \|x(t) - c\|^2 - r^2 = 0 \tag{3.44}$$

$$(\vec{s} + t\vec{d} - \vec{c}) \cdot (\vec{s} + t\vec{d} - \vec{c}) - r^2 = 0 \tag{3.45}$$

$$\begin{pmatrix} s_x + td_{x_{norm}} - c_x \\ s_y + td_{y_{norm}} - c_y \\ s_z + td_{z_{norm}} - c_z \end{pmatrix} \cdot \begin{pmatrix} s_x + td_{x_{norm}} - c_x \\ s_y + td_{y_{norm}} - c_y \\ s_z + td_{z_{norm}} - c_z \end{pmatrix} - r^2 = 0 \tag{3.46}$$

$$\begin{pmatrix} td_{x_{norm}} - (c_x - s_x) \\ td_{y_{norm}} - (c_y - s_y) \\ td_{z_{norm}} - (c_z - s_z) \end{pmatrix} \cdot \begin{pmatrix} td_{x_{norm}} - (c_x - s_x) \\ td_{y_{norm}} - (c_y - s_y) \\ td_{z_{norm}} - (c_z - s_z) \end{pmatrix} - r^2 = 0 \tag{3.47}$$

$$(td_{x_{norm}} - (c_x - s_x))^2 + (td_{y_{norm}} - (c_y - s_y))^2 + (td_{z_{norm}} - (c_z - s_z))^2 - r^2 = 0 \tag{3.48}$$

$$\begin{aligned} &\underbrace{(d_{x_{norm}}^2 + d_{y_{norm}}^2 + d_{z_{norm}}^2)}_{a} t^2 + \\ &\underbrace{-2((c_x - s_x)d_{x_{norm}} + (c_y - s_y)d_{y_{norm}} + (c_z - s_z)d_{z_{norm}})}_{b} t + \\ &\underbrace{((c_x - s_x)^2 + (c_y - s_y)^2 + (c_z - s_z)^2 - r^2)}_{c} \end{aligned} = 0 \tag{3.49}$$

The previous expression is a simple second order polynomial, this means that there are 3 ray-sphere intersection scheme that can occur:

- polynomial has 0 real root: the ray x does not intersect the sphere at all
- polynomial has 1 real root t_0: the ray x only touches the surface of the sphere at coordinates $x(t_0)$, without going through
- polynomial has 2 real roots t_0 and t_1: the ray enters the sphere, pass through and then exits at another disctinct point, denoted by $x(t_0)$ or $x(t_1)$ depending on their sign

We must notice that, as we are looking for the sphere footprint, the only rays we will be interested in, will be the rays that intersect the sphere only once, without going through, which corresponds to a polynomial with 0 valued determinant. In addition to that, as we imposed that the sphere should lie between the source and the detector, not behind the source, we should check that the solution is positive.

The second order polynomial determinant, that we will later equate to zero reads:

$$\Delta_{conic} = b^2 - 4ac \tag{3.50}$$

$$\begin{aligned} & 2^2((c_x - s_x)d_{x_{norm}} + (c_y - s_y)d_{y_{norm}} + (c_z - s_z)d_{z_{norm}})^2 - \\ = \; & 4 \times (d_{x_{norm}}^2 + d_{y_{norm}}^2 + d_{z_{norm}}^2) \\ & ((c_x - s_x)^2 + (c_y - s_y)^2 + (c_z - s_z)^2 - r^2) \end{aligned} \tag{3.51}$$

We can see that this expression can be factored by 4 and also by $\frac{1}{K^2}$, where K is the constant presentend in section 3.3.3.5 if we consider now that:

$$d_{x_{norm}} = \frac{d_x}{K} \tag{3.52}$$

$$d_{y_{norm}} = \frac{d_y}{K} \tag{3.53}$$

$$d_{z_{norm}} = \frac{d_z}{K} \tag{3.54}$$

We can then simplify the expression of the discrimant, in order to get the following conic, and replacing the ray direction term by the linear combination of u and v detector coordinate found earlier thanks to the projection matrix:

$$\begin{aligned} \Delta_{conic} = \frac{4}{K^2} \times \; & ((c_x - s_x)d_x + (c_y - s_y)d_y + (c_z - s_z)d_z)^2 - \\ & ({d_x}^2 + {d_y}^2 + {d_z}^2) \times \\ & ((c_x - s_x)^2 + (c_y - s_y)^2 + (c_z - s_z)^2 - r^2) \end{aligned} \tag{3.55}$$

Where each of the d_x,d_y,d_z can be developed using the members of the matrix M presented in section 3.3.3.4 :

$$\Delta_{conic} = \frac{4}{K^2} \times \begin{array}{l} [(c_x - s_x)(M(0)u + M(1)v + M(2)) + \\ (c_y - s_y)(M(3)u + M(4)v + M(5)) + \\ (c_z - s_z)(M(6)u + M(7)v + M(8))]^2 - \\ [(M(0)u + M(1)v + M(2))^2 + \\ (M(3)u + M(4)v + M(5))^2 + \\ (M(6)u + M(7)v + M(8))^2] \times \\ [(c_x - s_x)^2 + (c_y - s_y)^2 + (c_z - s_z)^2 - r^2] \end{array} \tag{3.56}$$

Next, the terms u and v, which are the main unknowns of our $2D$ conic equation can be factored:

$$\Delta_{conic} = \frac{4}{K^2} \times \begin{array}{l} [u \underbrace{((c_x - s_x)M(0) + (c_y - s_y)M(3) + (c_z - s_z)M(6))}_{\alpha_1} + \\ v \underbrace{((c_x - s_x)M(1) + (c_y - s_y)M(4) + (c_z - s_z)M(7))}_{\alpha_2} + \\ \underbrace{((c_x - s_x)M(2) + (c_y - s_y)M(5) + (c_z - s_z)M(8))}_{\alpha_3}]^2 - \\ [u^2 \underbrace{(M(0)^2 + M(3)^2 + M(6)^2)}_{\alpha_4} + \\ v^2 \underbrace{(M(1)^2 + M(4)^2 + M(7)^2)}_{\alpha_5} + \\ \underbrace{(M(2)^2 + M(5)^2 + M(8)^2)}_{\alpha_6} + \\ 2uv \underbrace{(M(0)M(1) + M(3)M(4) + M(6)M(7))}_{\alpha_7} + \\ 2u \underbrace{(M(0)M(2) + M(3)M(5) + M(6)M(8))}_{\alpha_8} + \\ 2v \underbrace{(M(1)M(2) + M(4)M(5) + M(7)M(8))}_{\alpha_9}] \\ \times \underbrace{((c_x - s_x)^2 + (c_y - s_y)^2 + (c_z - s_z)^2 - r^2)}_{\alpha_{10}} \end{array} \tag{3.57}$$

The identification process yields the following expression:

$$\Delta_{conic} = \frac{4}{K^2} \times \begin{array}{l} u^2\alpha_1^2 + v^2\alpha_2^2 + \alpha_3^2 + \\ 2uv\alpha_1\alpha_2 + 2u\alpha_1\alpha_3 + 2v\alpha_2\alpha_3 - \\ (u^2\alpha_4\alpha_{10} + v^2\alpha_5\alpha_{10} + \alpha_6\alpha_{10} + \\ 2uv\alpha_7\alpha_{10} + 2u\alpha_8\alpha_{10} + 2v\alpha_9\alpha_{10}) \end{array} \tag{3.58}$$

Which gives, with a proper factorization:

$$\Delta_{conic} = \frac{4}{K^2} \times \begin{array}{l} u^2(\underbrace{\alpha_1^2 - \alpha_4\alpha_{10}}_{A}) + v^2(\underbrace{\alpha_2^2 - \alpha_5\alpha_{10}}_{B}) + \\ u(\underbrace{2(\alpha_1\alpha_3 - \alpha_8\alpha_{10})}_{D}) + v(\underbrace{2(\alpha_2\alpha_3 - \alpha_9\alpha_{10})}_{E}) + \\ uv(\underbrace{2(\alpha_1\alpha_2 - \alpha_7\alpha_{10})}_{C}) + (\underbrace{\alpha_3^2 - \alpha_6\alpha_{10}}_{F}) \end{array} \tag{3.59}$$

3.5.2.2 Establishing conic section equation

We recognize in the previous equation the expression of a conic section. This can be seen more clearly by factoring all term using u^2, v^2, cross term uv and u,v and constant terms together, while equating the determinant, up to its scaling factor to 0:

$$\Delta_{conic} = Au^2 + Bv^2 + Cuv + Du + Ev + F = 0 \tag{3.60}$$

That can be rewritten using a symmetric matrix :

$$Au^2 + Bv^2 + Cuv + Du + Ev + F = 0 \tag{3.61}$$

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix}^{\intercal} \begin{pmatrix} A & C/2 & D/2 \\ C/2 & B & E/2 \\ D/2 & E/2 & F \end{pmatrix} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = 0 \tag{3.62}$$

or, equivalently :

$$\begin{pmatrix} u \\ v \end{pmatrix}^{\intercal} \begin{pmatrix} A & C/2 \\ C/2 & B \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} + \begin{pmatrix} D \\ E \end{pmatrix}^{\intercal} \begin{pmatrix} u \\ v \end{pmatrix} + F = 0 \tag{3.63}$$

$$x_0{}^{\intercal} A_0 x_0 + B_0{}^{\intercal} x_0 + C_0 = 0 \tag{3.64}$$

We have now, a fully developed conic expression, and we will be able to derive its properties from very well known technics.

3.5.2.3 Ellipse characterization

First of all, to identify, the type of the conic section, and, in the relevant case of an ellipse, determine its center, axis direction and vertices, we will be interested in factoring it in the normal form.

A simple rule will help us to know wether the conic is an ellipse, or if the angle between the projection plane normal, and the rays coming from the source is too important, hence generating an unbounded footprint. This property can simply be checked by computing the determinant of the A_0 matrix shown earlier:

$$\Delta_{A_0} = A \times B - \frac{1}{4} \times C^2 \tag{3.65}$$

If $\Delta_{A_0} > 0$ we have an ellipse, otherwise the projection of the sphere is more likely a parabola.

To go further on determining the two axis of the ellipse, we need a more simple conic expression, without cross terms in uv, this could be obtained using the following expression:

$$x_1{}^\intercal A_1 x_1 + B_1{}^\intercal x_1 + C_1 = 0 \tag{3.66}$$

where $x_1 = \begin{pmatrix} u_1 \\ v_1 \end{pmatrix}$ a rotated version (isometry) of the original u and v pixel detector coordinates such that A_1 a diagonal matrix.
A simple diagonalisation of A_0 will help us to achieve this goal:

$$x_0{}^\intercal A_0 x_0 + B_0{}^\intercal x_0 + C_0 = 0 \tag{3.67}$$

$$x_0{}^\intercal P_{diag} A_{diag} {P_{diag}}^{-1} x_0 + B_0{}^\intercal x_0 + C_0 = 0 \tag{3.68}$$

The previous equation is useful in our case because we know that A_0 is a real symmetric matrix, hence diagonalizable with real eigenvalues, and that its eigenvectors form an orthonormal basis.
As a direct consequence here, assuming we have normalized eigenvectors, we can write $P_{diag}^{-1} = {P_{diag}}^\intercal$. Let's see how this property is useful in our problem:

$$x_0{}^\intercal P_{diag} A_{diag} P_{diag}^{-1} x_0 + B_0{}^\intercal x_0 + C_0 = 0 \tag{3.69}$$
$$(P_{diag}^\intercal x_0)^\intercal A_{diag} (P_{diag}^\intercal x_0) + B_0{}^\intercal x_0 + C_0 = 0 \tag{3.70}$$
$$(P_{diag}^\intercal x_0)^\intercal A_{diag} (P_{diag}^\intercal x_0) + B_0{}^\intercal P_{diag} (P_{diag}^\intercal x_0) + C_0 = 0 \tag{3.71}$$
$$x_1{}^\intercal A_1 x_1 + B_1{}^\intercal x_1 + C_1 = 0 \tag{3.72}$$

This expression, with A_1 a diagonal matrix, $B_1 = P_{diag}^\intercal B_0$, and $C_1 = C_0$ holds for the following variable change: $x_1 = P_{diag}^\intercal \cdot x_0$.

Fortunately, finding the eigenvalues λ_1 and λ_2 and the corresponding eigen vectors V_1 and V_2 of a 2×2 matrix of the form $\begin{pmatrix} A & C/2 \\ C/2 & B \end{pmatrix}$ is easy, see for instance the fast diagonalization method exposed in [**kronenburg2013method**].

Let's define the following variables:

- trace of A_0: $Tr_{A_0} = A + B$
- difference of diagonal terms: $Td_{A_0} = A - B$
- determinant of A_0 is $\Delta_{A_0} = Td_{A_0}^2 + C^2$

The eigenvalues are simply the roots of the quadratic form derived from the $A_0 - \lambda$ matrix:

$$\lambda_1 = \frac{Tr_{A_0} + sign(Td_{A_0})\sqrt{\Delta_{A_0}}}{2} \tag{3.73}$$
$$\lambda_2 = \frac{Tr_{A_0} - sign(Td_{A_0})\sqrt{\Delta_{A_0}}}{2} \tag{3.74}$$

There are actually two equivalent solutions that allow to find the corresponding eigenvectors ${V_1}^0$ and ${V_2}^0$, the implementation performance will depend on the computer architecture. The implementation choice will rely on the availability of specific function units computing *arctan*, *cos* and *sin*, as well as the ability to handle branch efficiently in the code:

Solution 1: Without trigonometric functions In this case, we can use the closed form solution of the eigen vector problem to get:

$$P_{diag} = \left(\vec{V_1^0} \vec{V_2^0}\right) \tag{3.75}$$

where

$$\begin{cases} V_{1n}^0 = \begin{pmatrix} C/2 \\ \lambda_1 - A \end{pmatrix}, V_1^0 = \frac{V_{1n}^0}{\|V_{1n}^0\|}, & \text{if } C \neq 0 \\ V_1^0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, & \text{otherwise} \end{cases} \tag{3.76}$$

$$\begin{cases} V_{2n}^0 = \begin{pmatrix} C/2 \\ \lambda_2 - A \end{pmatrix}, V_2^0 = \frac{V_{2n}^0}{\|V_{2n}^0\|}, & \text{if } C \neq 0 \\ V_2^0 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, & \text{otherwise} \end{cases} \tag{3.77}$$

Solution 2: Using trigonometric functions In this case, we can use a simple identification technique, based on the fact that all 2×2 unitary matrix can be written as:

$$P_{diag} = \left(\vec{V_1^0}\vec{V_2^0}\right) = \begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{pmatrix} \tag{3.78}$$

The members of the matrix can be obtained by identifying θ:

$$\theta = \begin{cases} \frac{arctan\left(\frac{C}{Td_{A_0}}\right)}{2} & \text{if } Td_{A_0} \neq 0 \\ arctan2\left(\left(V_1^0\right)_1, \left(V_1^0\right)_0\right), \text{ with } V_1^0 \text{obtained using previous solution} & \text{otherwise} \end{cases} \tag{3.79}$$

We have now found the rotation matrix that allows us to express our conic section without cross terms, using $A_1 = \begin{pmatrix} A_1(0) & 0 \\ 0 & A_1(3) \end{pmatrix}$ and $B_1 = \begin{pmatrix} B_1(0) \\ B_1(1) \end{pmatrix}$, we can now write:

$$x_1{}^\intercal A_1 x_1 + B_1{}^\intercal x_1 + C_1 = 0 \tag{3.80}$$

$$A_1(0){u_1}^2 + A_1(3){v_1}^2 + B_1(0)u_1 + B_1(1)v_1 + C_1 = 0 \tag{3.81}$$

3.5.2.4 Ellipse expression in normal form

In order to study the ellipse properties, like its center, its semi-axis direction, length, area, eccentricity,... it is interesting to derive the normal form of the ellipse. To do so, we have to complete the square of its equation, this simple process here leads to (first step):

$$A_1(0){u_1}^2 + A_1(3){v_1}^2 + B_1(0)u_1 + B_1(1)v_1 + C_1 = 0 \tag{3.82}$$

$$A_1(0)\left(\underbrace{{u_1}^2}_{a^2}\underbrace{-2\left(\frac{-B_1(0)}{2\times A_1(0)}\right)u_1}_{-2ab}+\underbrace{\left(\frac{-B_1(0)}{2\times A_1(0)}\right)^2}_{+b^2}\right)+ \tag{3.83}$$

$$A_1(3)\left(\underbrace{{v_1}^2}_{a^2}\underbrace{-2\left(\frac{-B_1(1)}{2\times A_1(3)}\right)v_1}_{-2ab}+\underbrace{\left(\frac{-B_1(1)}{2\times A_1(3)}\right)^2}_{+b^2}\right) = -C_1+\frac{B_1(0)^2}{4\times A_1(0)}+\frac{B_1(1)^2}{4\times A_1(3)} \tag{3.84}$$

$$\tag{3.85}$$

Which, after proper factorization gives:

$$A_1(0)\left(u_1-\frac{-B_1(0)}{2\times A_1(0)}\right)^2 + A_1(3)\left(v_1-\frac{-B_1(1)}{2\times A_1(3)}\right)^2 = -C_1+\frac{B_1(0)^2}{4\times A_1(0)}+\frac{B_1(1)^2}{4\times A_1(3)} \tag{3.86}$$

We can now make the parallel with the simple "normal" ellipse equation using the following scheme:

$$a(x-b)^2 + c(y-d)^2 = e \tag{3.87}$$

$$\frac{(x-b)^2}{e/a} + \frac{(y-d)^2}{e/c} = 1 \tag{3.88}$$

where we have u_1 and v_1 represented as x and y, and the following constant identities:

$$a = A_1(0) \tag{3.89}$$

$$b = \frac{-B_1(0)}{2\times A_1(0)} \tag{3.90}$$

$$c = A_1(3) \tag{3.91}$$

$$d = \frac{-B_1(1)}{2\times A_1(3)} \tag{3.92}$$

$$e = -C_1+\frac{B_1(0)^2}{4\times A_1(0)}+\frac{B_1(1)^2}{4\times A_1(3)} \tag{3.93}$$

$$\tag{3.94}$$

Using the equation 3.5.2.4, and the derivation exposed in 3.5.2.3 it is now easy to compute:

- **The center of the ellipse:** Applying the inverse of the diagonalizing rotation, we get $\begin{pmatrix} u_c \\ v_c \end{pmatrix} = P_{diag} \begin{pmatrix} b \\ d \end{pmatrix}$. It should be noticed that the projection of the center of the sphere does not necessarily lines up with the center of the ellipse
- **The direction of the ellipse axis:** They are the vectors directly readable from the column vector of the unitary matrix P_{diag}: $\vec{V_1^0}$, which is the unit vector, associated with the greatest eigenvalue of the ellipse matrix, hence the smallest semi-axis. $\vec{V_2^0}$ being the unit vector standing for the largest semi-axis.
- **The length of the ellipse semi-axis:** They can be found thanks to the factorization step presented in equation 3.5.2.4, the one corresponding to the smallest axis is $l_0 = \sqrt{e/a}$ and the other corresponding to the largest axis in the P_{diag} matrix is $l_1 = \sqrt{e/c}$
- **The focal length:** The distance between the two foci can be easily obtained using the length of the ellipse semi axis as $f = \sqrt{l_1^2 - l_0^2}$
- **Eccentricity:** The eccentricity simply reads $\frac{f}{l_1}$
- **Area:** The area of the ellipse reads $\pi l_0 l_1$

3.5.3 Bounding box of the blob footprint

At this point, we have perfectly characterized the semi-major axis, and the semi-minor axis of the ellipse. From this we can easily derive the bounding box of the ellipse on the detector, i.e the rectangle A,B,C,D in pixel based coordinates. We figured this bounding box in green, on the figure 3.17.

It is also interesting to notice on the figure 3.17 that there are 2 other enclosing boxes that can be derived: the axis aligned bounding box of the ellipse, that will be derived in the next section 3.5.3.1, and the axis aligned bounding box of the $ABCD$ rectangle itself. The later is easy to compute with a set of min and max operator over the coordinates of the $ABCD$ vertices, but it has the inconvenient of overestimating the total blob projection area, hence leading to an unnecessary high computational burden.

Another completely different approach would be to compute the AABB[9] of the three dimensional AABB of the truncated blob function itself. This process would simply amount to project 8 points onto the detector, and then find the minimum bounding box of those points. This approach also lead to an overestimation of the AABB of the blob projection, but

[9] *Axis Aligned Bounding Box*

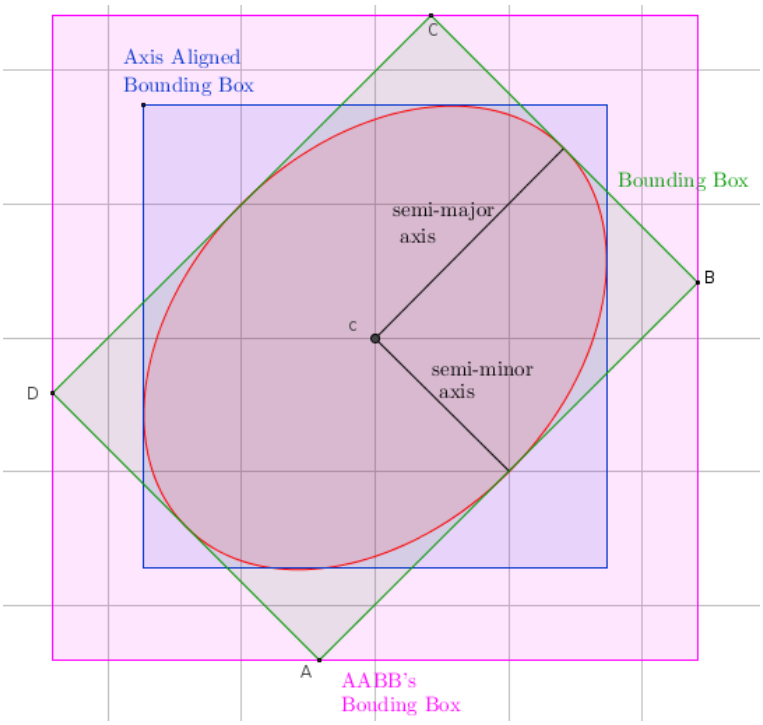


Figure 3.17: Various Bounding Boxes that can be used to enclose an ellipse

its simplicity may be of interest if the important amount of overlap between blob footprint is not an issue.

3.5.3.1 Axis Aligned Bounding Box of an ellipse

It can be seen from figure 3.17, that iterating over each pixel that is part of the bounding box of the projected sphere will not be an easy task, because its profile can be skewed, hence requiring multiple box intersection calculation. The AABB however provides a simple way to perform a loop over the 2-dimensional set of pixels that intersect the sphere footprint.

Finding the AABB of an ellipse is fortunately also a simple task, considering that the locus of an ellipse can be defined using a parametric equation of 1 variable t, based on its normal definition seen in equation 3.5.2.4:

$$\left(\frac{x_1 - b}{l_0}\right)^2 + \left(\frac{y_1 - d}{l_1}\right)^2 = 1 \tag{3.95}$$

$$cos(t)^2 + sin(t)^2 = 1 \tag{3.96}$$

$$\tag{3.97}$$

Where we have made the variable change $x_1 - b = l_0 cos(t)$ and $y_1 - d = l_1 sin(t)$.

We also recall that we have made a first change of variable $x_1 = P_{diag}^{\intercal} \cdot x_0$, in equation 3.5.2.3, where P_{diag} was a unitary matrix. This allow us to rewrite the parametric equation in function of the initial variables:

$$El(t) = P_{diag}\left(\begin{pmatrix} b \\ d \end{pmatrix} + \begin{pmatrix} l_0 cos(t) \\ l_1 sin(t) \end{pmatrix}\right) \tag{3.98}$$

$$= \begin{pmatrix} u_c \\ v_c \end{pmatrix} + \begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{pmatrix} \begin{pmatrix} l_0 cos(t) \\ l_1 sin(t) \end{pmatrix} \tag{3.99}$$

$$= \begin{pmatrix} u_c \\ v_c \end{pmatrix} + \begin{pmatrix} l_0 cos(\theta)cos(t) - l_1 sin(\theta)sin(t) \\ l_0 sin(\theta)cos(t) + l_1 cos(\theta)sin(t) \end{pmatrix} \tag{3.100}$$

As the ellipse is bounded, and is a convex geometrical object, we will be able to find for each coordinate, the parameter t for wich the partial derivative $\frac{\partial El(t)_u}{\partial t}$ and $\frac{\partial El(t)_v}{\partial t}$ along both axis vanish:

Axis u Lets derive the optimal coordinates for the u direction on the detector

$$0 = \frac{\partial El(t)_u}{\partial t} \tag{3.101}$$

$$= -l_0 cos(\theta)sin(t) - l_1 sin(\theta)cos(t) \tag{3.102}$$

$$-l_0 cos(\theta)sin(t) = l_1 sin(\theta)cos(t) \tag{3.103}$$

$$\frac{sin(t)}{cos(t)} = -\frac{l_1 sin(\theta)}{l_0 cos(\theta)} \tag{3.104}$$

$$\Leftrightarrow t = \begin{cases} t_{u1} = arctan\left(-\frac{l_1}{l_0} tan(\theta)\right) \\ t_{u2} = \pi + arctan\left(-\frac{l_1}{l_0} tan(\theta)\right) \end{cases} \tag{3.105}$$

Reinjecting those solutions into the parametric model gives us the following optimal values for the u axis:

$$opt_{u1} = u_c + l_0 cos(\theta)cos(t_{u1}) - l_1 sin(\theta)sin(t_{u1}) \tag{3.106}$$

$$opt_{u2} = u_c + l_0 cos(\theta)cos(t_{u2}) - l_1 sin(\theta)sin(t_{u2}) \tag{3.107}$$

hence

$$min_u = min(opt_{u1}, opt_{u2}) \tag{3.108}$$

$$max_u = max(opt_{u1}, opt_{u2}) \tag{3.109}$$

Axis v Lets derive the optimal coordinates for the v direction on the detector

$$
\begin{aligned}
0 &= \frac{\partial El(t)_v}{\partial t} && (3.110)\\
&= -l_0 sin(\theta)sin(t) + l_1 cos(\theta)cos(t) && (3.111)\\
-l_0 sin(\theta)sin(t) &= -l_1 cos(\theta)cos(t) && (3.112)\\
\frac{sin(t)}{cos(t)} &= \frac{-l_1 cos(\theta)}{-l_0 sin(\theta)} && (3.113)\\
tan(t) &= \frac{l_1}{l_0}\frac{1}{tan(\theta)} && (3.114)\\
\Leftrightarrow t &= \begin{cases} \begin{cases} t_{v1} &= \pi/2 \\ t_{v2} &= 3\pi/2 \end{cases} & \text{if } tan(\theta) = 0 \\ \begin{cases} t_{v1} &= arctan\left(\frac{l_1}{l_0}\frac{1}{tan(\theta)}\right) \\ t_{v2} &= \pi + arctan\left(\frac{l_1}{l_0}\frac{1}{tan(\theta)}\right) \end{cases} & \text{otherwise} \end{cases} && (3.115)
\end{aligned}
$$

Reinjecting those solution into the parametric model gives us the following optimal values for the u axis:

$$
\begin{aligned}
opt_{v1} &= v_c + l_0 sin(\theta)cos(t_{v1}) + l_1 cos(\theta)sin(t_{v1}) && (3.116)\\
opt_{v2} &= v_c + l_0 sin(\theta)cos(t_{v2}) + l_1 cos(\theta)sin(t_{v2}) && (3.117)
\end{aligned}
$$

hence

$$
\begin{aligned}
min_v &= min(opt_{v1}, opt_{v2}) && (3.118)\\
max_v &= max(opt_{v1}, opt_{v2}) && (3.119)
\end{aligned}
$$

3.5.4 Splatting the blob

3.5.4.1 Abel transform of a blob over its footprint

Thanks to the previous section, we know exactly how to compute the AABB of a sphere in an arbitrary projective geometry, our only remaining task is now to compute the Abel transform, exposed in section 2.2.7, for every ray hitting the detector bins inside the AABB. To do so, the only thing we need is the distance between the blob center, and its orthogonal projection onto the ray under consideration, as figured on the scheme 3.18

As stated earlier, spherically symmetric functions value only depends on the distance r to the center c, and so do their Abel transform. In the framework of CBCT projective geometry,

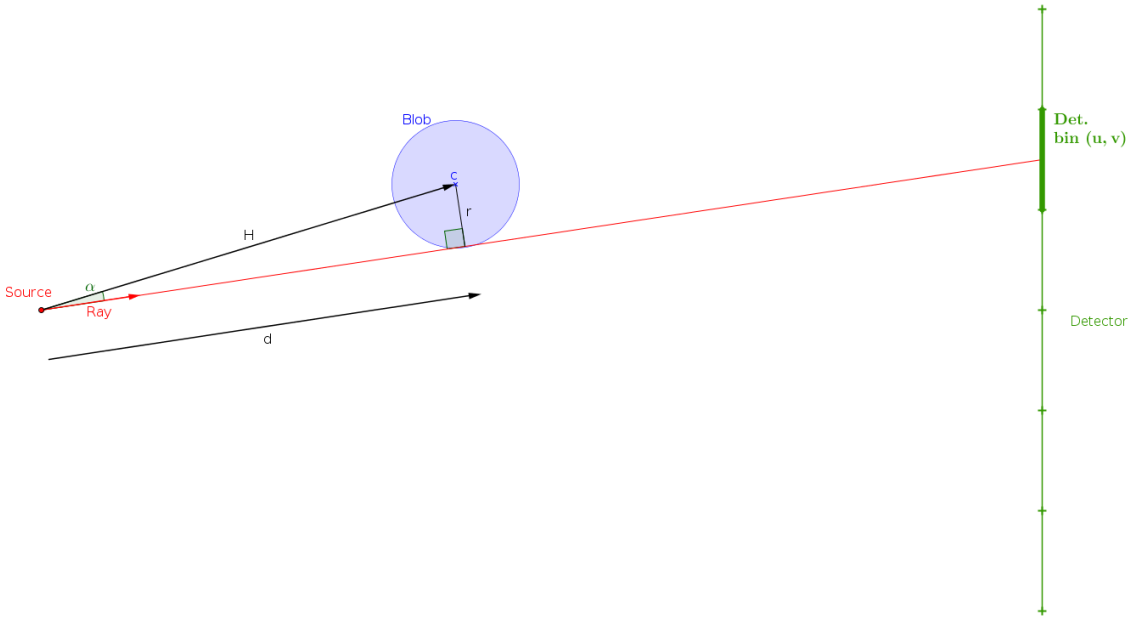


Figure 3.18: Profile of a blob volume element crossed by a ray

we have shown in section 3.3.3.5 how to derive the ray direction vector $\frac{\vec{d}}{\|\vec{d}\|}$ that hit the detector bin at coordinates (u, v).

Deriving the distance r for any ray will be an easy task using simple geometry, given that we have, following the pythagore theorem: $r^2 = \|\vec{H}\|^2 - \|\vec{d}\|^2$, and the following relationship between d and H:

$$\|\vec{d}\| = \frac{\vec{d}}{\|\vec{d}\|} \cdot \vec{H} \tag{3.120}$$

$$\tag{3.121}$$

Reinjecting this expression inside the pythagorean relationship gives

$$r^2 = \|\vec{H}\|^2 - \|\vec{d}\|^2 \tag{3.122}$$

$$= \|\vec{H}\|^2 - \left(\frac{\vec{d}}{\|\vec{d}\|} \cdot \vec{H}\right) \tag{3.123}$$

$$\Leftrightarrow r = \sqrt{\|\vec{H}\|^2 - \left(\frac{\vec{d}}{\|\vec{d}\|} \cdot \vec{H}\right)} \tag{3.124}$$

3.5.4.2 Experiments

Blob bounding box estimation We implemented our blob based projector, in order to validate our model on a simple geometrical setting, that we figured on scheme 3.19 :

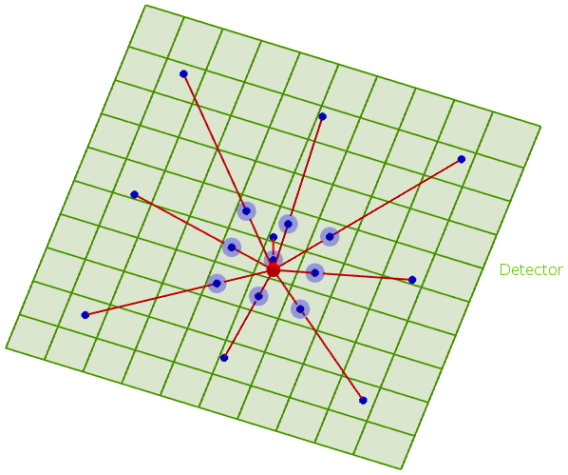


Figure 3.19: Geometrical setting for our blob footprint evaluation test

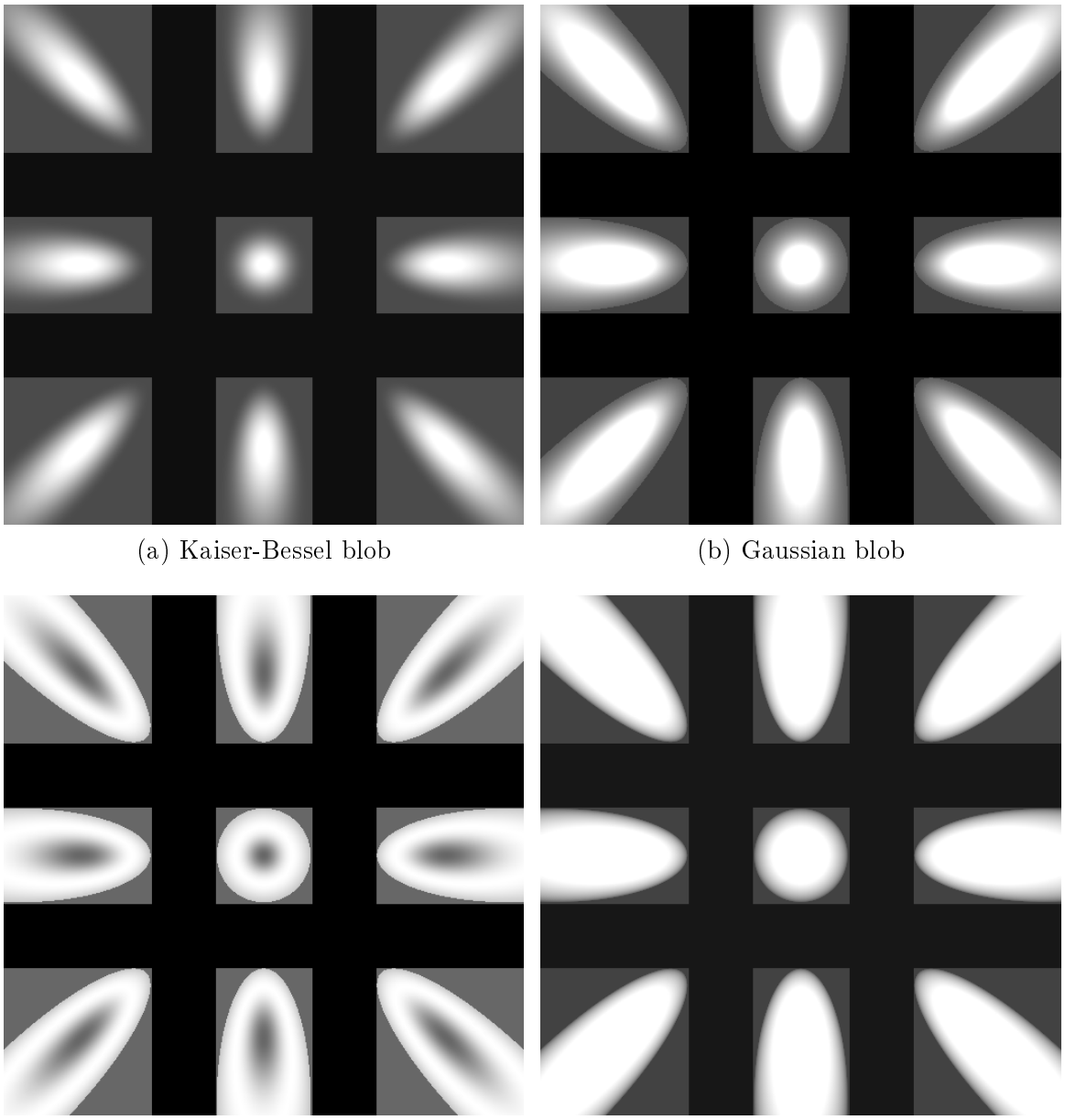
This geometrical setting voluntarily features an extremely important cone angle, in order to highlight the eccentricity of the ellipse, and the role of the bounding box. It also features axis aligned, as well as non axis aligned blobs footprints, in order to check for the robustness of our method when rotation angles have a remarkable value, ie $0,\pi/2,\pi,...$

We used this geometric setup to project various radially symmetric functions defined in the section 2.2.7, and exposed the results on figure 3.20

Ellipse characterization in a realistic geometry In order to assess the relevance of our model in the framework of a realistic geometric model, we used a simple model similar to the default geometry defined in RTK[10], see [**rit2014reconstruction**]. The flat panel is modeled as a 256×256 mm detector with a resolution of 512×512 pixels, with a pixel size of $500 \times 500 \mu m$. The image plane, where the sphere to be projected will be located, is at minimum 1 meter away from the X-Ray source. The detector is 1.536m away from the source, resulting in a moderate magnification ratio of about 1.5. We figured this simple geometry on the figure 3.21. In order to quantify some metrics about the footprint, we decided to choose a sphere of radius 1 mm to model the "blob" that will be projected.

We were interested in quantifying the minimum and maximum eccentricity of the ellipsis resulting from the sphere projection, according to their position on the detector. We reported the eccentricity map on figure 3.22 as well as a drawing of a unit ellipse with the minimum and maximum eccentricity, found for the geometry exposed in figure 3.21.

[10] *Open Reconstruction ToolKit*



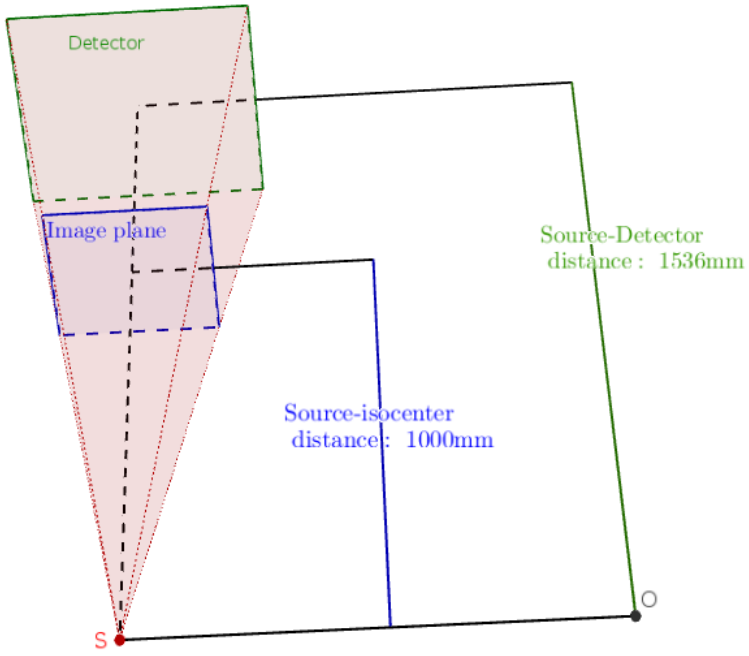
(a) Kaiser-Bessel blob (b) Gaussian blob

(c) Mexican Hat blob (d) Plain sphere indicator

Figure 3.20: Projection of various volume elements in projective geometry

In order to get an idea of the computational burden of the projection of a "blob" onto a detector, and the difference between footprint size for various position onto the detector, we reported the total ellipse area in number of pixels for every position on the detector on figure ??.

Ellipse characterization in a more divergent geometry We decided to assess the behaviour of the blob footprint, in a geometry where the cone angle is larger, and the mag-



(a)

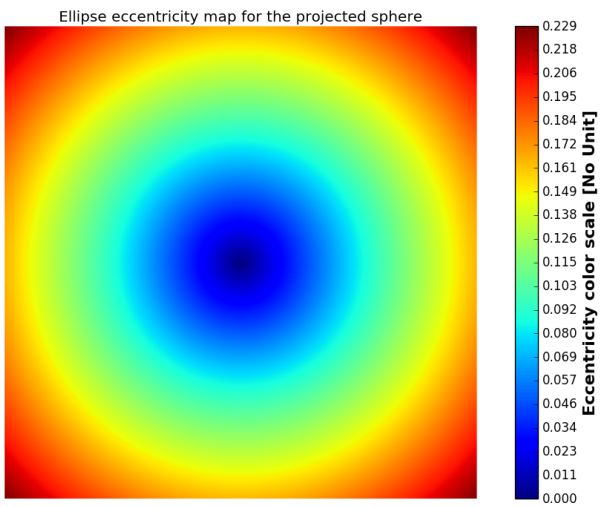
Figure 3.21: Realistic geometry for a CBCT system

nification ratio in higher, close to 3 in this case. The radius of the sphere however were kept identical in this experiment (1mm). We figured the geometry on figure 3.23

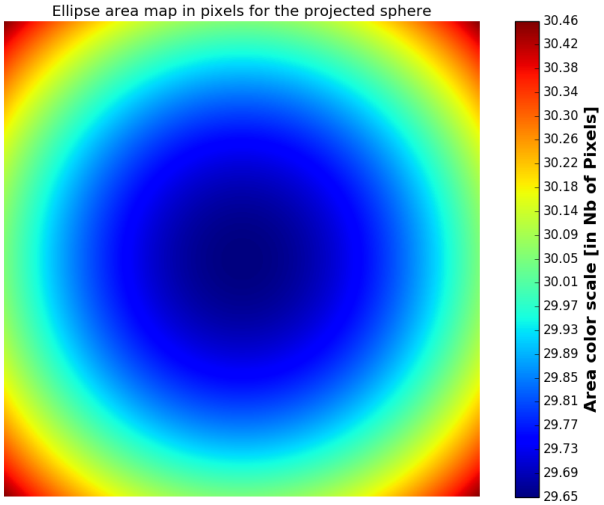
We also assessed the eccentricity map in this geometry, as well as the ellipse size, in terms of number of pixels, on figure 3.24

Discussion Is is really interesting to see that the form of the projected ellipse, denoted by its eccentricity does not experience large variations in our two test cases presented on figure 3.22 3.24 from 0.23 to 0.44. In practice, the ellipse generated is visually very close to a circle. Althoug we did not figured the Abel transform for these profile, one can infer that the profiles won't experience a large deviation from the axis aligned case. Those observations led us to consider that the separable blob projection model exposed in [**ziegler2006efficient**], that assumed a cylindrical axis aligned detector would probably map nicely to the case of a flat panel detector. This observation also comfort the relevance of the separable spline driven model proposed in [**momey2013spline**], that used a slightly different geometrical framework to derive the ellipse parameters.

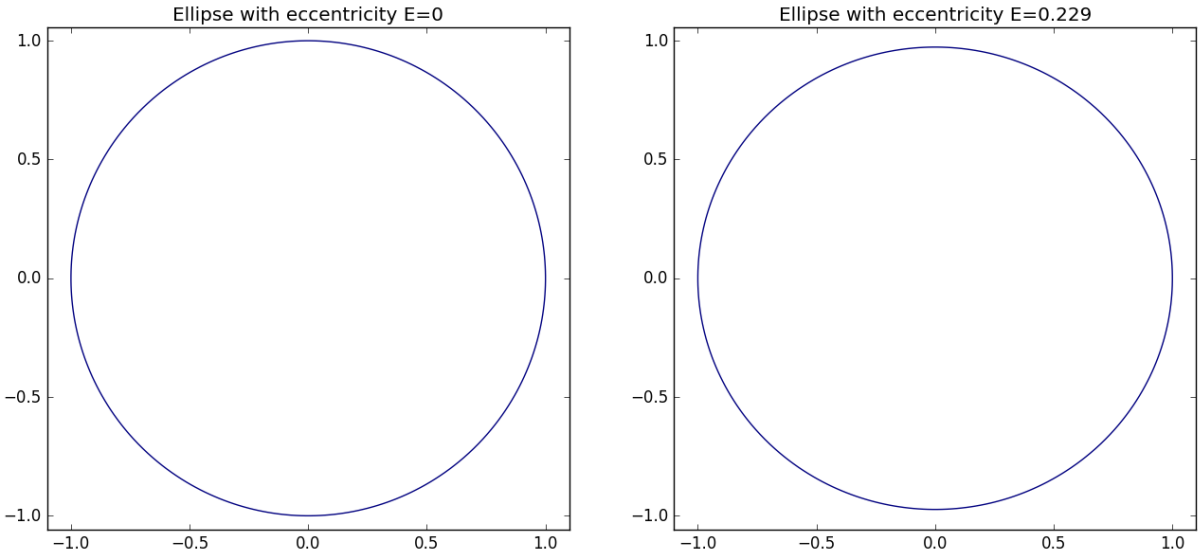
The blob projection framework presented here has the advantage of being drawn directly from the projection matrix elements, thus being adaptable to arbitrary volume discretization scheme, whithout any modifications. Unfortunately, we did not had enough time to implement this method in a proper reconstruction framework to challenge its performances on practical reconstruction tasks.



(a) Ellipse eccentricity map



(b) Sphere footprint size, in number of pixels



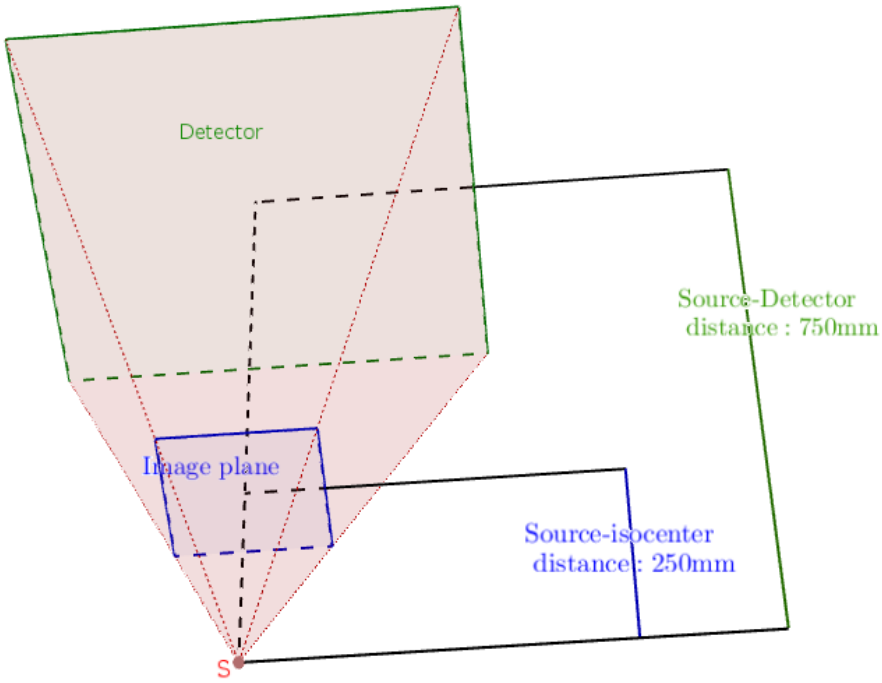
(c) Minimum and maximum eccentricity featured in the projection

Figure 3.22: Sphere footprint eccentricity

3.6 Conclusion

In this chapter, we have seen that multiple strategies have been derived to model the tomographic operator, related to data discretization scheme, interpolation models, and implementation target.

One of the most successful approach for fast tomographic operators implementation in the recent years was the GPU based implementation. However, apart from the matrix-vector case, we have shown that a host-based memory model was not, as is a simple strategy to set-up,

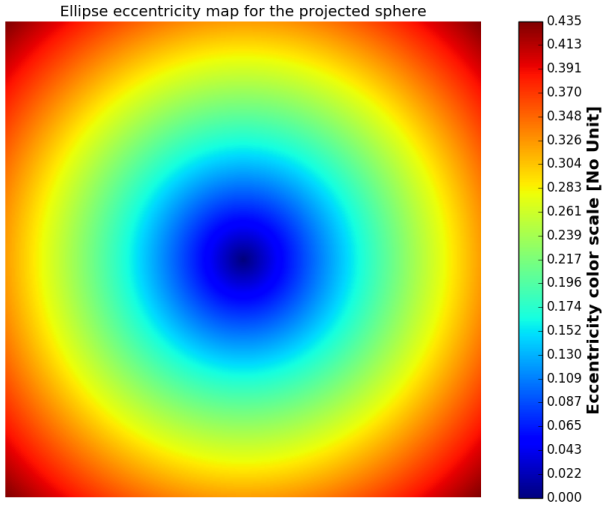


(a)

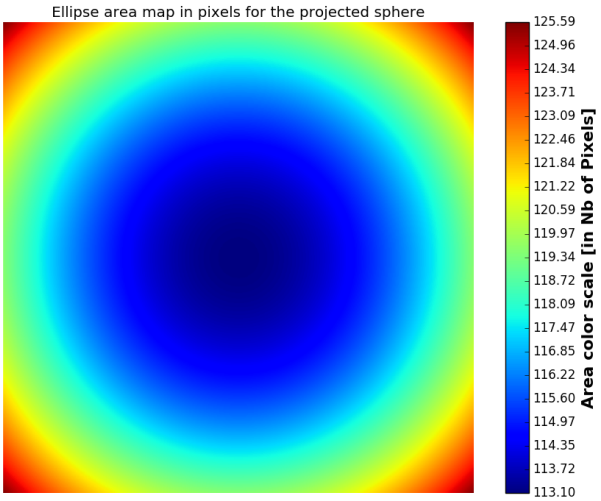
Figure 3.23: Cone Beam geometry with a wider angle

in order to obtain good performances. However, we also gave an insight about the increasing programmability of GPUs, through the use of some modern C++ features, that may in a near future, along with new interconnection technology, allow for more complex operation to take place on the GPU as well.

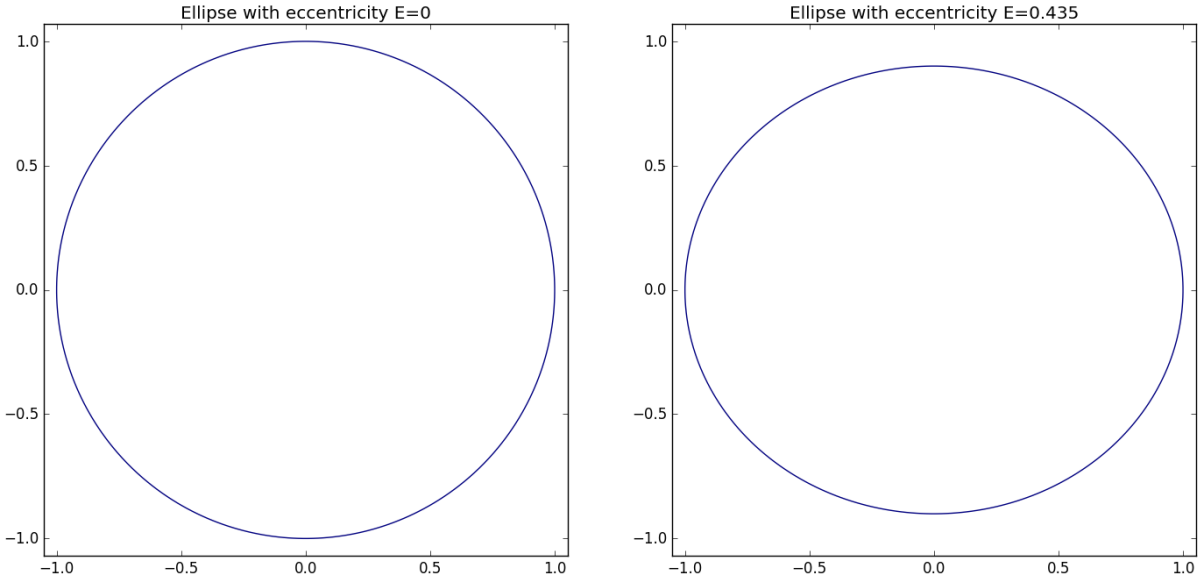
We developped a blob based projection model, natively compliant with flat panel detectors and CBCT geometry described as projection matrices. However we found-out that the discrepancy between, our simple discrete model, and the separable model presented in a previous work [**ziegler2006efficient**], was not really significant: ellipse eccentricity of 0.435 in the worst case scenario.



(a) Ellipse eccentricity map



(b) Sphere footprint size, in number of pixels



(c) Minimum and maximum eccentricity featured in the projection

Figure 3.24: Sphere footprint eccentricity

Chapter 4: First Order Methods applied to Tomography

Sommaire

4.1 Introduction

As seen in the previous chapter, many aspects of signal processing, especially those related to tomography, or challenging signal retrieval, amounts to solve a linear system or optimize a cost function over a high dimensional space.

Throughout this chapter, we will only consider the category of convex objective functions to be optimized over convex sets like $\mathbb{R}^n$, and we will restrict our study only to a small subset of the large number of existing optimization methods, in order to concentrate on the simple ones that can be easily applied on large problems.

We propose here to study those well known algorithms in the context of cone beam tomography, which practically differs from the context of a generic linear problem in the sense that in this case linear operators are generally computed on the fly, which basically limits the available operations to matrix vector (forward projection), or adjoint matrix vector product (backprojection). It is also worth noting that the design of tomographic linear operators is intrinsically linked to the physical acquisition process and the data discretization model which we have seen in chapter 2, as well as the choices made towards a computationally efficient implementation, as seen in chapter 3.

In particular, in many cases in cone beam tomography, forward and backward projection operator may not be adjoints of each other, and feature slightly different properties from a signal procesing point of view, regarding aliasing for instance. This discrepancy may actually result in new derivations of known algorithms having slightly different behaviour, depending on the tomographic operators properties in the framework of linear algebra, weither some of these discrepancies can be interpreted as implicit preconditioning or not.

In a first section, we will address tomogaphic reconstruction as a linear problem, and recall when such problems can be considered as easy or difficult to solve, from a theoretical point of view. We will also make a few remarks and recall a few results from the compressive sensing

theory, in order to highlight the fact that this sampling theory mainly relies on properties from the field of linear algebra, instead of functional analysis.

In the next section, we will mostly focus on the linear least square, which is a quadratic problem that have been extensively studied for centuries, and is still an active area of research, and in particular we will give an insight about first order methods used to solve this problem.

Finally, we will use a simple proximal splitting framework, in order to explain how a primal/dual based optimality certificate can be derived for the least square.

4.2 Linear equality constraints - Solving $M\vec{x} - \vec{y} = 0$

4.2.1 Definitions

Here we have:

- $\vec{x} \in \mathbb{R}^k$ an unknown vector
- $\vec{y} \in \mathbb{R}^n$ a data vector
- $M \in \mathbb{R}^{n \times k}$ a matrix

4.2.2 What are linear equalities

Let's begin with one of the first historically studied problem in linear algebra, solving the equality $M\vec{x} - \vec{y} = 0$. Although being of a moderate interest in engineering, this formulation has been used to model the problem of tomographic algebraic reconstruction problem during the 70's , see for instance [**gordon1970algebraic**].
But this equality also arise in other fields, its role might reduce to define a convex set. A very good introduction about this topic can be found in chapter 1 of [**boyd2004convex**]. A solution of a set of linear equality, called an "'affine set"' indeed provide a stable space where any kind of affine combinations amounts to a feasible solution of the the initial linear equality.

Another way of seeing a linear equality, that allows reasoning on geometry, is to say that the set of solutions of each linear equality defines an hyperplane in $\mathbb{R}^k$, and that the set that satisfies all equalities is the intersections of all hyperplanes.

4.2.3 Is the problem hard ?

Simple linear algebra gives us some hints about the difficulty of the problem: first, simply by studying the shape of M, and using some simple algebra properties like equality of row and column rank, and rank nullity theorem:

- If $k > n$, there is more unknown than known data, following the rank nullity theorem, the solution is an affine set of dimension $k - \text{rank}(M)$ which cannot be zero because $\text{rank}(M) \leq n < k$. In this case, solving the problem would lead to an infinite set of solution which is not always useful for common numerical problems.

- If $k < n$ the problem is said to be overdetermined, there are more constraints than unknown, this time the solution is an affine set of dimension $k - \text{rank}(M)$ with $\text{rank}(M) \leq k < n$. The solution may be unique if there are exactly $n - k + 1$ equivalent constraints in the problem, if more constraints are equivalents, we return to the infinite set of solution, and otherwise the system is inconsistent and has no solutions. In real cases, if there are smalls numerical errors on the system model M or if $\vec{y}$ comes from a noisy measurement process, the probability that multiple rows are perfect multiples of each other is extremely low, and the problem is more likely to be inconsistent.

We can now try to analyze what happen for a square problem, when $n = k$, we will use the determinant to give a more geometric interpretation of the simple rank nullity theorem, :

If the matrix M is square, we can theoretically derive its determinant. As seen in section 2.2.4, a nice geometric interpretation of the determinant helped us to link its value with the volume of the hyper-parallelogram in $\mathbb{R}^k$ defined by the k columns vectors of M. If the family formed by those vectors is not free, then the hyper-parallelogram collapses into an only $\text{rank}(M)$ dimensional figure and then its hyper-volume in dimension k amounts to zero.

In the case the determinant in non-zero, M is invertible, then there is a unique $\vec{\hat{x}}$ that verify $M\vec{\hat{x}} - \vec{y} = 0$

In the case of a null determinant, M is said to be singular, and there are two cases that can be diagnosed using the rank of M, exactly as seen previously, and giving a more geometrical interpretation to the value of $\vec{y}$:

- $\vec{y} \notin Im(M)$ then it means that the set of hyperplanes may contains parallel and non coincident elements making the matrix and its corresponding system respectively under determined and non consistent. As all hyperplans never cross in a single point, the system has no solution.

- $\vec{y} \in Im(M)$ then underlying system of equation is also under-determined because of parallel hyperplans, but admits in fact a set of solutions of the from $\vec{x_0} + \vec{v}, \vec{v} \in \ker(M)$ where $\vec{x_0}$ can be any solution point in $\mathbb{R}^k$ and $\vec{v}$ is any vector of $\mathbb{R}^k$ such that $M\vec{v} = \vec{0}_{\mathbb{R}^n}$ i.e that would not influence the property of $\vec{x_0}$ being a solution, because it will be sent to $\vec{0}$ in $\mathbb{R}^n$ by M. Following the rank nullity theorem, the solution has dimension

$k - \text{rank}(M)$.

It is important to notice here the role of $\vec{y}$, that can change the possible set of solution from a void set to an infinite set, and possibly high dimensional linear space, we will talk about this specific aspect a bit later.

Unfortunately, in signal processing, $\vec{y}$ is often a noisy data vector, and M is a large scale matrix, with possibly millions of entry that precludes us from using any determinant based method for inverting M or even assessing feasibility of the problem, that would anyway probably be inconsistent.

4.2.4 What happen in real numerical cases ?

As seen previously, data inconsistency may lead to a problem without solution in many cases, but it is important to understand why those inconsistency arise, how their effect can be smoothed using some block/iterative algorithm.

Let's review a simple study case, illustrated on figure 4.1 and 4.2.

For instance some of the row vectors $\vec{a_0}$, $\vec{a_1}$, $\vec{a_2}$ defined by M could be nearly collinear, but the hyperplanes b_0, b_1, b_2 defined by the $\vec{a_i}$ and especially their corresponding values in $\vec{y}$, by adding a bias in the codimension could make them to cross a third hyperplane at two points P and Q far away from each other.

Through these example we can see that a measurement matrix M with a maximum of nearly orthogonal measurment row vectors, having similar norms will yield somehow more manageable linear systems, where we can, for instance relax some constraints if the problem is overdetermined, or have the guarantee to make small errors in the solution if the system is full rank, but has "noise" in the measure $\vec{y}$.

4.2.5 How hard is the problem ?

This simple observation made in the previous section has been formalized through the notion of "Condition Number" C_M which quantifies how much the solution will vary in the worst case if we make an error $\vec{e}$ in the measurement $\vec{y}$ in $\mathbb{R}^n$, for a given matrix M.

A local approximation of the relative error norm in the solution reported to the relative norm of the error in the measure space, for a given norm $\|\|_u$ is given by:

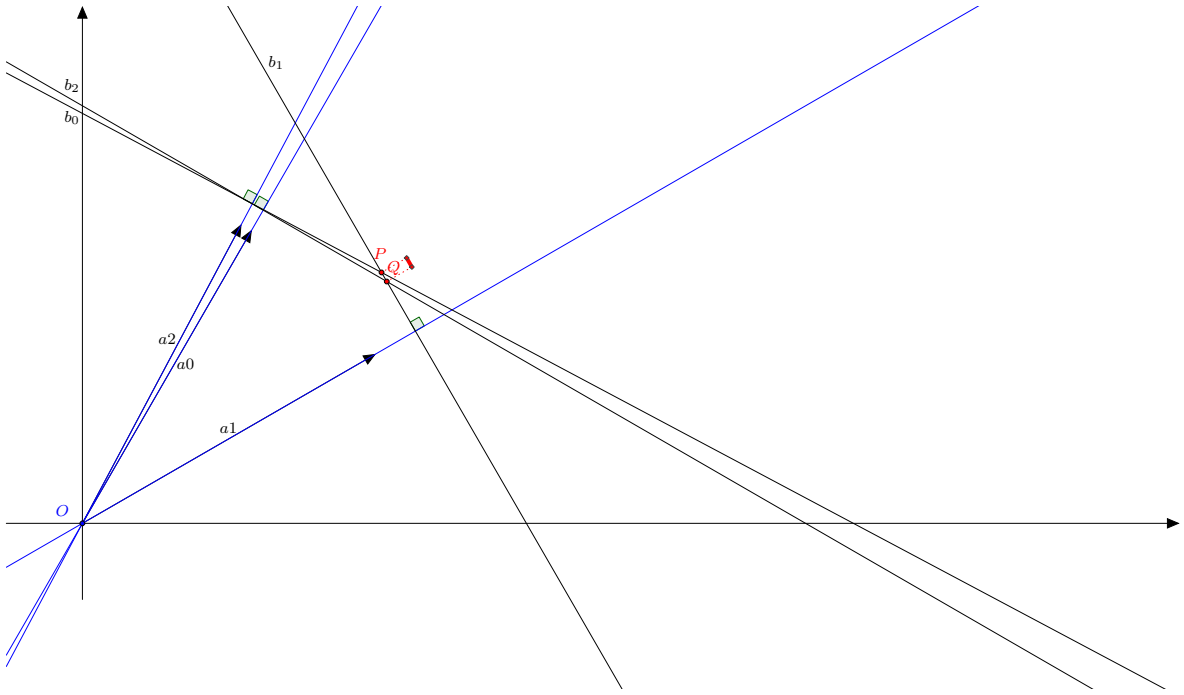


Figure 4.1: The case where row vectors of M: $\vec{a_0}$, $\vec{a_1}$, $\vec{a_2}$ are not really colinear and yield manageable inconsistency

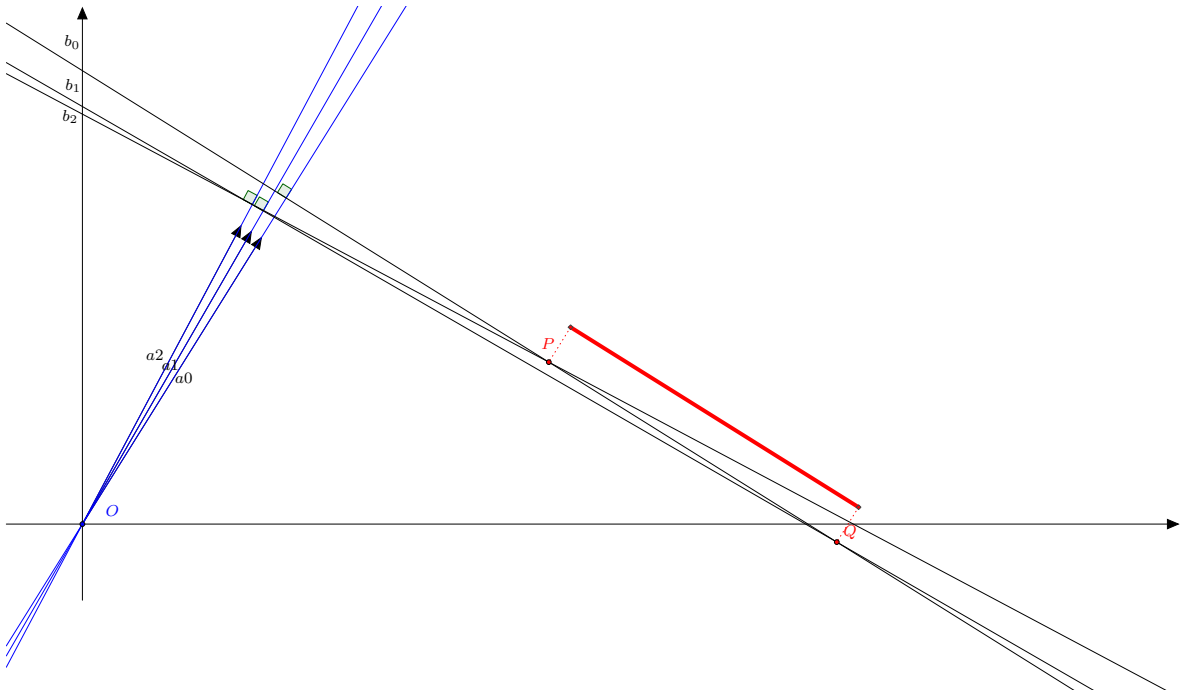


Figure 4.2: The case where row vectors of M: $\vec{a_0}$, $\vec{a_1}$, $\vec{a_2}$ are nearly colinear and yield important inconsistency

$$C_M = \frac{\|M^{-1}\vec{e}\|_u / \|M^{-1}\vec{y}\|_u}{\|\vec{e}\|_u / \|\vec{y}\|_u} \tag{4.1}$$

$$= \frac{\|M^{-1}\vec{e}\|_u}{\|\vec{e}\|_u} \cdot \frac{\|\vec{y}\|_u}{\|M^{-1}\vec{y}\|_u} \tag{4.2}$$

This error depends on the coordinates of $\vec{y}$ in $\mathbb{R}^n$, and of the direction of the error $\vec{e}$, wich will respond differently according to the geometry of M. We can bound the expression in eq 4.1 in the worst case. To do so, we should consider the case where the measure vector $\vec{y}$, when projected by M will experience a scaling that will reduce its norm to the minimum, and simultaneously the error $\vec{e}$ will experience the maximum possible scaling for a vector in $\mathbb{R}^n$:

$$C_M = \max_{\vec{e}\neq 0} \frac{\|M^{-1}\vec{e}\|_u}{\|\vec{e}\|_u} \times \max_{M^{-1}\vec{y}\neq 0} \frac{\|\vec{y}\|_u}{\|M^{-1}\vec{y}\|_u} \tag{4.3}$$

In the following steps, we will be using $u = 2$ as the l_2 norm for expressing the condition number. We also recall the definition of the operator norm, for a non-singular matrix M:

$$\|M\| = \max_{x\neq 0} \quad \frac{\|Mx\|}{\|x\|} \tag{4.4}$$

$$= \max_{\|x\|=1} \quad \|Mx\| \tag{4.5}$$

and we will also be interested in using M to express the operator norm of $\|M^{-1}\|$ that cannot always be computed easily.

$$\|M^{-1}\| = \max_{x\neq 0} \quad \frac{\|M^{-1}x\|}{\|x\|} \tag{4.6}$$

$$= \max_{My\neq 0} \quad \frac{\|y\|}{\|My\|} \quad \text{with } y = M^{-1}x \tag{4.7}$$

$$\frac{1}{\|M^{-1}\|} = \min_{y\neq 0} \quad \frac{\|My\|}{\|y\|} \tag{4.8}$$

$$= \min_{\|y\|=1} \quad \|My\| \tag{4.9}$$

which helps to express condition number directly in terms of operator norm

$$C_M = \|M^{-1}\|\|M\| \tag{4.10}$$

Moreover, it is interesting to see that both operator bounds can be derived easily for the case of the l^2 norm, whithout having to compute the inverse of M: maximizing the squared

l_2 norm of the M-projection under the constraint that x should be a unit vector, yields the following lagrangian optimization problem:

$$\|M\|^2 = \min_{\|x\|=1} \quad \|Mx\|^2 \tag{4.11}$$

$$= \max_{x} \quad \min_{\lambda \geq 0} \quad x^\intercal M^t M x - \lambda(1 - x^\intercal x) \tag{4.12}$$

Which, differentiated in the variable x gives the following critical point:

$$M^t M x = \lambda x \tag{4.13}$$

Where we can clearly identify the lagrangian multiplier λ as an eigenvalue of $M^t M$, that should be the largest one : $\lambda_{M^t M \quad max}$ if we want to satisfy the optimality condition. The exact same scheme can be derived for the squared matrix norm $\|M^{-1}\|^2$:

$$\frac{1}{\|M^{-1}\|^2} = \min_{\|y\|=1} \quad \|My\| \tag{4.14}$$

$$= \min_{y} \quad \max_{\lambda \geq 0} \quad y^\intercal M^t M y - \lambda(1 - y^\intercal y) \tag{4.15}$$

When differentiated in y, the critical point also verifies $M^t M y = \lambda y$, where λ is an eigenvalue of $M^t M$, but this time, the optimality holds when it is the lowest : $\lambda_{M^t M \quad min}$.

The condition number for M can then be easily derived from the eigendecomposition of $M^t M$:

$$C_M = \sqrt{\frac{\lambda_{M^t M \quad max}}{\lambda_{M^t M \quad min}}} \tag{4.16}$$

The first remark we can make is that the optimal condition number is 1 and it is obtained for all isometries. In practical point of view, one can notice that maximum and minimum eigenvalues can be computed through the use of power methods, as stated in [**zeng2000unmatched**].

Link with Hadamard definition of well posed problems In the case where one consider solving $M\vec{x} - \vec{y} = 0$ with M a square matrix, it is interesting to take a look at how the condition number C_M of M gives an insight about wether the problem is well posed according to the definition given by Jacques Hadamard, based on 3 properties :

- ***A solution exists:*** Use the determinant to prove that the solution exists

- ***The solution is unique:*** Here again, a non-zero determinant is sufficient to prove that the solution is unique
- ***A least square solution exists:*** the condition number must exist, ie, M^tM should not feature zero valued eigenvalues
- ***The least square solution is unique*** : nullspace of M, should be of dimension 0, hence M should be of rank k hence it should not feature zero eigenvalues as in the previous point.
- ***The solution's behavior changes continuously with the initial conditions:*** the condition number tells us how an error $\vec{e}$ in the dataset $\vec{y}$ will affect relatively the solution in the worst case. [**chretien2014perturbation**] Weyl's inequality.

4.2.6 What problems should be considered as easy ?

Introduction According to what we have seen, isometry are the linear transform that yields the more stable inverses, and it can be proved that in common Hilbert spaces of finite dimension, any linear isometry is an orthogonal transformation: Let's take $x, y \in \mathbb{R}^k$, and $\langle ., . \rangle$ be the common dot product in $\mathbb{R}^k$

$$\|x+y\|^2 - \|x-y\|^2 = \|M(x+y)\|^2 - \|M(x-y)\|^2 \tag{4.17}$$
$$\langle x+y, x+y\rangle - \langle x-y, x-y\rangle = \langle Mx+My, Mx+My\rangle - \langle Mx-My, Mx-My\rangle \tag{4.18}$$
$$4x^\intercal y = 4x^\intercal M^\intercal My \tag{4.19}$$
$$\langle x, y\rangle = \langle x, M^\intercal My\rangle \tag{4.20}$$
$$\tag{4.21}$$

Using this equality assigning to x successively all the vector of the basis of our hilbert space give use that $M^\intercal M = Identity$, so M is an unitary matrix, which by definition is orthogonal.

Relaxation of isometry and orthogonality property : RIP It is interesting to notice that a famous relaxation of the isometry property has been defined in [**candes2005decoding**] : the restricted isometry property, which helped to derive a lot of recovery guarantees in the field of compressive sensing.

It characterize the $n \times k$ matrices M, which can be considered as a collection of column vectors $(v_j)_{j\in J=0,1,\dots,k-1} \in \mathbb{R}^n$ for which all $n \times |T|$ submatrices M_T, formed by a subset $T \subset J$ of the columns of M of cardinality $|T| \leq s$, with $1 \leq s \leq k$ are close to an isometry, for the real coefficients $(y_j)_{j\in T}$ relatively to a constant δ_s :

$$(1-\delta_s)\|y\|_2^2 \leq \|M_T y\|_2^2 \leq (1+\delta_s)\|y\|_2^2 \tag{4.22}$$

Where the minimum value of δ_s for which this inequality holds is called the restricted isometric constant

Similarly, the S, S'-restricted orthogonality constant $\theta_{S,S'}$ for $S + S' \leq |J|$ is defined as he smallest quantity such that:

$$|\langle M_T y, M_{T'} y' \rangle| \leq \theta_{S,S'} . \|y\| \|y'\| \tag{4.23}$$

holds for all disjoints sets $T, T' \subseteq J$ of cardinality $|T| \leq S$ and $|T'| \leq S'$

In [**james2014eigenvalues**], the author recall various results that arose from the compressive sensing framework, such that the sufficient δ_s for which convex or greedy relaxation of the $l0$ sparsity promoting algorithms can achieve perfect recovery.

Although verifying the RIP property over a given matrix is a NP-Hard problem, it as been shown that independant identically gaussian, Bernouilli and partial fourier matrices features the RIP property with exponentially high probability.

The case of Gaussian matrices In the framework of random matrices theory, a very interesting study of the condition number of the specific case of Wishart matrices $G(n, n) = XX^*$ where X is a $n \times m$ i.i.d gaussian matrix with mean 0 and finite variance have been performed in [**edelman1988eigenvalues**]. The author has proved, that the distribution of their condition number, relatively to the size n of the matrix, κ/n converges pointwise to the following probability distribution function:

$$\frac{2x + 4}{x^3} e^{-2/x - 2/x^2} \tag{4.24}$$

More precisely, when the distribution of the i.i.d gaussian is of mean μ and variance σ, the expression $\frac{2\sigma}{\mu} \kappa n^{3/2}$ also convergespointwise towards 4.24

And the expectation of the log condition number for these matrices is

$$E(log\kappa) = log(n) + c + o(1) \tag{4.25}$$

with $c \approx 1.537$

More recently, in the framework of compressive sensing and restricted isometry property, it has been, recalled in [**candes2005decoding**] and [**james2014eigenvalues**] that, given a $n \times k$ gaussian matrix with mean zero and variance $1/p$, and a subset T of its columns, we have, for large values of n and fixed T:

$$1 - \delta(M_T) \leq \lambda_{min}(M_T^\intercal M_T) \leq \lambda_{max}(M_T^\intercal M_T) \leq 1 + \delta(M_T) \tag{4.26}$$

where $\delta(M_T) \approx 2\sqrt{|T|/n} + |T|/n$

This inequality has been generalized in a probabilistic manner for all subsets T where $|T| \leq s$ as follows:

$$f(r) = \sqrt{k/n}.(\sqrt{r} + \sqrt{2H(r)}) \tag{4.27}$$

$$P(1 + \delta_s > [1 + (1 + \epsilon)f(r)]^2) \leq 2.e^{-kH(r).\epsilon/2} \tag{4.28}$$

With H the entropy function : $H(q) = -qlog(q) - (1-q)log(1-q)$, with$0 < q < 1$.

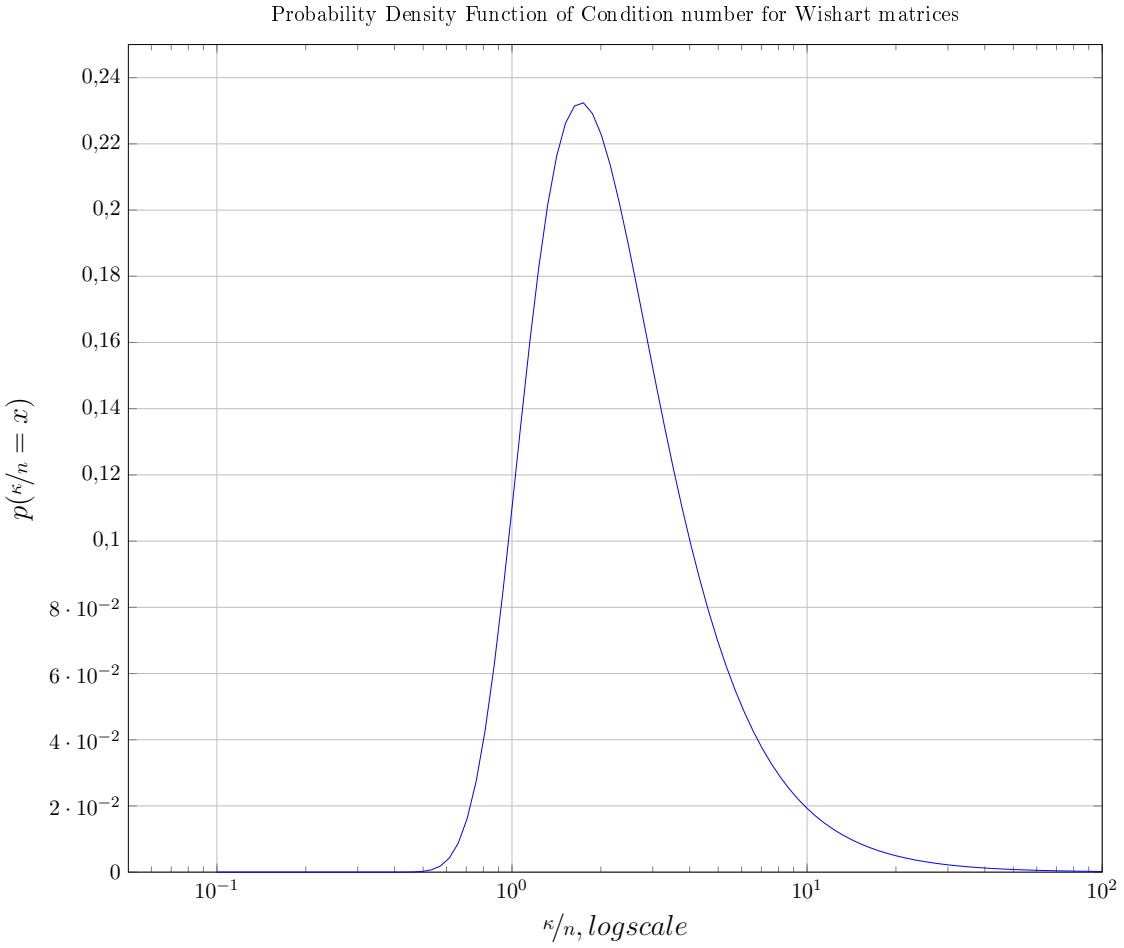


Figure 4.3: Probability distribution function of the condition number of a Wishart matrix of size n, with mean 0 and finite variance

Mutual coherence Although RIP emphasize on the isometry property for the subset of sparse vectors and is very difficult to prove, the recent concept of mutual coherence provides a somehow more geometric intuition, related to the independance of the variables in the

measurment vectors. In our understanding, this concept allows to extend the notion of rank in a more practical way. Indeed, in the field of numerical analysis, the cases where numerically colinear vectors are found in a matrix does not occurs very often, hence leading to some cases where rank and condition number can be counterintuitively correlated (high rank matrices with extremely large condition number). This is not intrinsically a problem, but mutual coherence concept filled a gap in linear algebra tools in order to derive recovery results for instance in [**donoho2006stable**].

The mutual coherence of a matrix M is defined as the maximum absolute value of the cross-correlations between the columns of M: let $(v_j)_{j\in J=0,1,\ldots,k-1} \in \mathbb{R}^n$ be the column vectors of M, normalized such that $\langle v_i, v_i\rangle = 1$, the mutual coherence of M reads:

$$\underset{1\leq i\neq j`k}{max} |\langle v_i, v_j\rangle| \tag{4.29}$$

This concept was mainly developped in [**donoho2001uncertainty**] for studying the uncertainty principle stating that "if two bases are mutually incoherent, no nonzero signal can have a sparse representation in both bases simultaneously".

4.2.7 Algorithms

4.2.7.1 Introduction

There are numerous methods used to solve the problem $Mx = y$. Many of them adress the problem where M is a square matrix. We may however refer to the problem $Mx = y$ in all cases, because we can use the surrogate problem $M^tMx = M^ty$ as a square matrix problem.

Given the size of the matrix in our case, and the fact that we assumed that we will never write down the entire projection matrix, we will be forced to discard nearly all methods that does not only rely on matrix vector product of the form Mv or M^tv:

Cofactors method The inverse of the square matrix M can be directly expressed as the adjoint matrix of the cofactors weighted by the inverse of the determinant. However, as seen earlier, we won't be able to get the value of the determinant, and it will probably be equal to zero in many cases. Anyway we will consider that we are not necessarily interested in computing the matrix inverse for any point, but we would like to get the inverse for one single point at a time.

Gauss Jordan Elimination As we do not have a full expression of the matrix in memory, we won't be able to triangularize our system matrix, and perform backsubstitution.

LU, QR factorization For the same reason stated for the Gauss-Jordan elimination, we won't be able to use any forward/backward substitution method due to physical constraints of memory storage.

4.2.7.2 Fixed point iteration methods

Fixed point iteration method stand for a very large class of algorithm that aims at iteratively solving a fixed point problem of the form $AX = X$ with A a linear operator in ou case. This class of method may be of interest, because they allow to iteratively refine a solution from one iteration to the next. In the general case where our matrix M may have a bad conditionning, an iterative approach may be the right strategy

Jacobi method Jacobi method assumes that one can decompose the $n \times n$ matrix M into a sum of matrices $M = G + H$ where G is full rank, and easily invertible, for instance it is set to a diagonal matrix in the Jacobi method.

This decomposition leads us to a fixed point search in the high dimensional space $\mathbb{R}^n$:

$$Mx - y = 0 \tag{4.30}$$
$$Gx + Hx = \vec{y} \tag{4.31}$$
$$Gx = -Hx + \vec{y} \tag{4.32}$$
$$x = -G^{-1}Hx + G^{-1}\vec{y} \tag{4.33}$$
$$x = M'x + \vec{y'} \tag{4.34}$$

Where $M' = -G^{-1}H$ and $\vec{y'} = G^{-1}\vec{y}$.

This fixed point search can be carried out using a simple iterative scheme

$$\vec{x^{k+1}} = M'\vec{x^k} + \vec{y'} \tag{4.35}$$

According to the Picard fixed point theorem, this class of algorithm converges if the operator $T : \vec{x} \to M'\vec{x} + \vec{y'}$ is a strict contraction, i.e if there exists $\rho \in [0, 1[$ such that

$$\forall (x, x') \in \mathbb{R}^n \times \mathbb{R}^n, x \neq x' : \quad \|Tx - Tx'\| < \rho \|x - x'\| \tag{4.36}$$
$$\|M'x + \vec{y'} - M'x' - \vec{y'}\| < \rho \|x - x'\| \tag{4.37}$$
$$\|M'x - M'x'\| < \rho \|x - x'\| \tag{4.38}$$
$$\|M'(x - x')\| < \rho \|x - x'\| \tag{4.39}$$
$$\|M'\| \|(x - x')\| < \rho \|x - x'\| \tag{4.40}$$
$$\|M'\| < \rho \tag{4.41}$$

This condition can be interpreted as a $[0, 1[$-Lipshitz continuity property of the linear application M', which is equivalent to a constraint over the operator norm of M', that reduces to a constraint over its largest singular value: $0 \leq \sigma(M')_{max} < 1$.

Preconditionning interpretation The generic matrix decomposition scheme $M = G + H$ presented in the previous paragraph can also be viewed as a preconditionning problem:

$$M\vec{x} - \vec{y} = 0 \tag{4.42}$$
$$G\vec{x} + H\vec{x} = \vec{y} \tag{4.43}$$
$$G\vec{x} = -H\vec{x} + \vec{y} \tag{4.44}$$
$$\vec{x} = -G^{-1}H\vec{x} + G^{-1}\vec{y} \tag{4.45}$$
$$(G^{-1}H + Id)\vec{x} = G^{-1}\vec{y} \tag{4.46}$$
$$G^{-1}(H + G)\vec{x} = G^{-1}\vec{y} \tag{4.47}$$
$$G^{-1}M\vec{x} = G^{-1}\vec{y} \tag{4.48}$$
$$\tag{4.49}$$

The action of casting the equivalent problem $G^{-1}M\vec{x} = G^{-1}\vec{y}$ instead of $M\vec{x} = \vec{y}$ is called preconditionning. This method is actually useful when the condition number of $G^{-1}M$ is lower than the condition number of M, and of course when G is easily invertible.

Other approaches The Gauss-Seidel method can be view as an extension of Jacobi method, with a different matrix decomposition of the form $M = D + (L + U)$ where, L is a strictly lower triangular matrix, D is a non singular diagonal matrix, and U is a stricly upper triangular matrix. Both Jacobi and Gauss-Seidel iterates can be modified in order to ensure and/or accelerate convergence, in the case where the matrix iterate have a poor Lipshitz constant. Among those methods we can cite the weighted Jacobi iterations, and the famous successive over-relaxation based methods (SOR, SSOR) which introduced the more general idea of overrelaxtion to iterative methods. Those kind of approach has seen promising developement recently, known as Scheduled Relaxation Jacobi method see [**adsuara2016scheduled**], however Jacobi method basically relies on a decomposition that will not be suitable for our problem, again, due to practical issues.

4.2.8 Projection onto convex sets

As seen in the section 4.2.2, we recall that the non-empty set of solution solution of a set of linear equality $Mx = y$, is a convex set $C_{M,y}$.

4.2.8.1 Convex set and proximal mapping

In the following developements, we will be using the framework of monotone operator theory in Hilbert spaces, the interested reader may refer to [**bauschke2011convex**]. In the framework of convex analysis, we can provide a convex, lower semi continuous cost function $\delta_{C_{M,y}}$ that stands for the solutions of the linear equality $Mx = y$, using the indicator function of the convex set $C_{M,y} : x \in \mathbb{R}^n \quad \text{s.t} \quad Mx = y$:

$$\delta_{C_{M,y}}(x) \equiv \begin{cases} 0 & \text{if } x \in C_{M,y} \\ +\infty & \text{otherwise} \end{cases} \tag{4.50}$$

In the framework of proximal mapping operators, we also recall that we can define the proximity operator of the function $\delta_{C_{M,y}}(x)$ as:

$$prox_{\gamma\delta_{C_{M,y}}}(z) = \underset{x}{argmin} \quad \frac{1}{2}\|x - z\|_2^2 + \gamma\delta_{C_{M,y}}(x) \tag{4.51}$$

$$= Proj_{C_{M,y}}(z) \tag{4.52}$$

Where $Proj_{C_{M,y}}(z)$ is the operator that stands for the projection onto the convex set $C_{M,y}$, but we will elaborate more on this topic in the next chapter 5.

Once the projection operator is defined for the convex set, one can simply apply the most simple proximal algorithm, see [**parikh2014proximal**]: the proximal point method, that reads:

$$x^{k+1} = prox_{\delta_{C_{M,y}}}(x^k) \tag{4.53}$$

For $C_{M,y}$ a non-void set, the algorithm is trivial and only one iteration is needed. We conclude that all the whole method will rely on the projection operator that we will derive in the next section : 4.2.8.2.

4.2.8.2 Kaczmarz and POCS method

Geometric methods for solving linear system of equations encountered a limited success despite there fast convergence rate. Among those methods, we can cite the Kaczmarz approach, that

was used for tomography in the 70's, under the name ART.

The idea of the method is really simple, it aims at sequentially projecting a solution estimate over all n hyperplans defined by the $n \times k$ matrix M. More formally, the atomic update that has to be repeated reads as follows:

$$x^k = x^{k-1} - (\langle M_{(i,*)}, x^{k-1} \rangle - y_i) \frac{M_{(i)}}{\|M_{(i,*)}\|_2^2} \tag{4.54}$$

Where

- $M_{(i,*)} \in \mathbb{R}^k$ is the i^{th} row of the matrix M
- y_i is the i^{th} element of the vector y

This extremely simple update is indeed a projection, onto the convex set defined by $\langle M_{(i,*)}, x^{k-1} \rangle = y_i$, and as long as there is no rank deficiency in the matrix M, the sequential execution of all n projection defined in the matrix M for multiple iteration should make x^k to converge to a solution of $Mx = y$, hence yielding a proper POCS method.

Unfortunately, although this method is simple, all our remarks from section 4.2.4 still apply, and if the problem is hard, for instance it contains inconstistencies, the problem has no solution, and the method will not converge toward a good approximation satisfying most of the constraints.

It is also interesting to notice that the condition number value for the matrix M will play a role in this algorithm convergence. Indeed, if the condition number $\kappa(M) = 1$, M is an isometry, the method converges in only one iteration, yelding a perfect n-step projections method.

However, the more the hyperangle between the row vectors of the matrix M will be close to zero (collinearity), the slower the projection method will get, we give a simple illustration of this geometrical intuition on figure 4.4.

This problem is well known from the tomography community, which noticed that radiographic projections acquired physically close to each other, i.e whose gantry acquisition angular displacement was close to 0, usually featured sets of measurment vector whose hyperangle was also close to zero. To overcome this problem, various pojection ordering strategy have been studied, see [**hamaker1978angles**], [**herman1993algebraic**], [**guan1994projection**], [**mueller1997weighted**], [**kong2012evaluation**] and even more recently for the few view case : [**zheng2011identifying**].

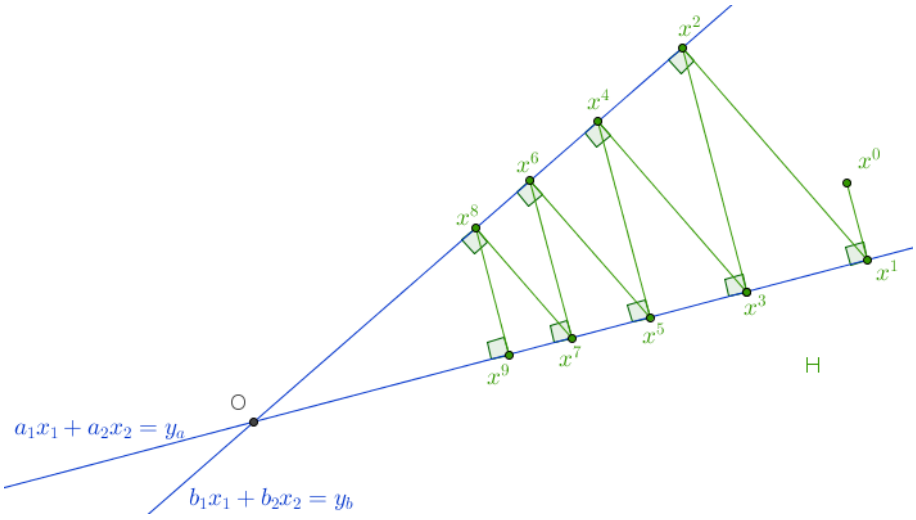


Figure 4.4: Alternative projections onto two convex sets in $\mathbb{R}^2$

4.2.8.3 Variation of the Kaczmarz method

Two major drawbacks of the Kaczmarz method, that preclude from using it in practical cases are:

- It is essentially a sequential method, that cannot be easily parallelized, which is problematic for large systems of equations.
- The method appeared to be quite sensible to inconsistencies in the data, especially when M has a large condition number.

In order to overcome this problems, many "smooth" variations of Kaczmarz method have been derived:

- smooth the individual hyperplan projection operation by using a relaxation factor $0 < \lambda \leq 1$ or $1 < \lambda \leq 2$ (symmetry).
- Compute the coordinates of the projection of the current solution on all hyperplans, and then assign the barycenter of this set of coordinates to the be the new solution estimate: this is the basic idea of Cimmino method, although it initially also involved a notion of symmetry.
- Only use a subset of the hyperplans, and apply the same strategy
- mix multiple of the above strategies

The basic idea of using the hyperplan projections barycenter can be formalized using the following iterates definition:

$$x^k = \frac{1}{n}\sum_{i=0}^{n-1} x^{k-1} - (\langle M_{(i,*)}, x^{k-1}\rangle - y_i)\frac{M_{(i,*)}}{\|M_{(i,*)}\|_2^2} \tag{4.55}$$

$$= x^{k-1} - \frac{1}{n}\sum_{i=0}^{n-1}(\langle M_{(i,*)}, x^{k-1}\rangle - y_i)\frac{M_{(i,*)}}{\|M_{(i,*)}\|_2^2} \tag{4.56}$$

It is clearly visible that all hyperplan projections and the volume correction can be computed in parallel, and that this operations involve two simple matrix-vector operation, so that this method has a simple matrix expression:

$$x^k = x^{k-1} - \frac{1}{n}M^t W(Mx - y) \tag{4.57}$$

Where W is a diagonal weighting matriw, where each $M_{i,i}$ contains the inverse of the squared L_2 norm of the i^{th} row of M : $W_{i,i} = \frac{1}{\|M_{(i,*)}\|_2^2}$.

It is interesting to see that this formulation can be seen as a variant of gradient descent, with an original preconditioner.

4.2.8.4 Simultaneous algebraic technic

We consider SIRT, see [**gilbert1972iterative**] and SART ([**andersen1984simultaneous**]) related methods to be slightly different from the hyperplan projection method seen in 4.2.8.3, as they do not account for a L_2 normalization of the measurment vectors $M_{(i)}$.

Instead, SART like algorithm update equation for a single voxel x_j of index j, and a subset of equation ranging from i_{beg} to $i_{end} \leq n$ (excluded) reads:

$$x_j^k = x_j^{k-1} - \frac{\sum_{i=i_{beg}}^{i_{end}-1} M_{i,j}\frac{\langle M_{(i,*)}, x^{k-1}\rangle - y_i}{\|M_{(i,*)}\|_1}}{\sum_{i=i_{beg}}^{i_{end}-1} M_{i,j}} \tag{4.58}$$

The SART update in equation 4.58, has to be performed for any arbitrary union of subsets that covers the whole set of equations in order to complete one iteration. It should be noticed that the previous update equation imply first a projection step, that can be carried out in parallel for every y_i, and then a backprojection step, that can be carried out in parallel for every x_j, making this algorithm particularly well suited for a parallel implementation, depending on the size of the chosen subset of equation. In the case where $i_{beg} = 0$ and $i_{end} = n$, there is only one subset, and the SART amounts to SIRT, its maximally parallel version, which however,

tends to feature a slower convergence. In his case, the update equation can be rewritten in matrix form:

$$x^k = x^{k-1} - W_1 M^t W_2 (Mx - y) \tag{4.59}$$

Where W_1 is the matrix of the form: $W_1 = \begin{pmatrix} \frac{1}{\|M_{(*,0)}\|_1} & 0 & \cdots & 0 \\ 0 & \frac{1}{\|M_{(*,1)}\|_1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\|M_{(*,k-1)}\|_1} \end{pmatrix}$
Whose role is to make $W_1 M^t$ a matrix whose rows sums to 1, assuming that M is non-negative.

And W_2 is the matrix of the form: $W_2 = \begin{pmatrix} \frac{1}{\|M_{(0,*)}\|_1} & 0 & \cdots & 0 \\ 0 & \frac{1}{\|M_{(1,*)}\|_1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\|M_{(n-1,*)}\|_1} \end{pmatrix}$
Whose role is to make $W_2 M$ a matrix whose rows sums to 1, assuming that M is non-negative.

Understanding the role of the W_1 relatively to the back projection operator M^t and the role of W_2 relatively to the projection operator M is fundamental in order to design a generic software able to handle all flavours of projection/backprojection operators, for the SART-like algorithms.

For instance, one can notice that in case the back-projection operator M^t features a simple interpolation sampling on the detector, then its equivalent matrix operator has a builtin row-wise normalization, such that W_1 is an identity matrix, and one save some computation and/or memory space.

Convergence analysis of the SART algorithm, as described in this section has been studied in [**jiang2003convergence**].

4.2.8.5 Going further with the randomized Kaczmarz method

It is interesting to observe that, driven by the need for fast optimization algorithm, capable of handling extremely large datasets, single sample or batch based optimization methods, like the one initially derived by Kaczmarz, recently regained the interest of mathematicians in the framework of stochastic optimization. Among the recent work derived from Kaczmarz/Cimmino methods, we can cite the randomized Kaczmarz method, see [**strohmer2009comments**], and the more general stochastic optimization method from Gower and Richtarik analyzed in [**gower2015randomized**].

4.3 Linear least square

In the following developments, as we will use less 2D geometry analogy, we will drop the $\vec{}$ superscript to denotes vectors. All lower case variables should be considered as vector, unless stated otherwise.

4.3.1 Introduction

We will now talk about one of the most common problem that arise in many fields of engineering : the least square problem, whose objective reads:

$$\underset{x\in\mathbb{R}^k}{argmin} \quad \frac{1}{2}\|Mx-y\|_2^2 \tag{4.60}$$

In the field of signal processing it can be seen as a signal retrieval problem where we have:

- $x \in \mathbb{R}^k$ the k dimensional unknown signal to be retrievied
- $y \in \mathbb{R}^n$ the n dimensional samples, generally equivalent to the following model of noisy measurements $y = Mx^* + \epsilon$, where the model of the noise, ϵ is random centered gaussian distribution with finite variance σ^2, a condition for the least square estimate to be efficient.
- $M \in \mathbb{R}^{n\times k}$ a linear measurements matrix, projecting the "'reality"' of the signal to be retrieved x^* in k dimensions, to a measurement space of dimension n, following a measurement model we wish to be as close as possible to the truth.

4.3.2 Least Square and Bayesian framework

This objective is very important in science because it has a simple interpretation in the Bayesian framework:
Let's recall that $y = Mx^* + \epsilon$ and that ϵ is a random process that follows an homoscedastic multivariate normal law: $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and we can write $y \sim \mathcal{N}(Mx^*, \sigma^2)$.

Now the Bayes theorem can be used to assess the probability of a candidate solution x, given a measure y:

$$p(x|y) = \frac{p(x) \times p(y|x)}{p(y)} \tag{4.61}$$

Where we have y the observation vector, whose probability $p(y)$ for which we have no statistical apriori, is supposed equiprobable, and, as such will be considered as a constant α_0.

Our current candidate solution x, for which we currently have no statistical a-priori, which equiprobability $p(x)$ over $\mathbb{R}^k$ can be modeled as a constant α_1. It can be noticed that researchers have used graph based statistical apriori, based on Random Markov Field model (RMF), to setup maximum a posteriori (MAP) strategy along with the Expectation Maximization (EM) algorithm in the past, see [**green1990bayesian**]
In our simple case, we can first establish a marginal version of the conditional probability $p(y_i|x)$, which, as seen in section 4.3.1 amounts to the following normal distribution:

$$p(y_i|x) = \frac{1}{\sqrt{2\pi\sigma^2}} exp^{-\frac{((Mx)_i - y_i)^2}{2\sigma^2}} \tag{4.62}$$

Using the fact that all y_i are independants with the same distribution, we can write

$$p(y|x) = \prod_{i=1}^{n} p(y_i|x) \tag{4.63}$$

Which, using the property of exponential, can be written in a vectorial fashion:

$$p(y|x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{\|Mx-y\|_2^2}{2\sigma^2}} \tag{4.64}$$

The likelihood of x then reads:

$$p(x|y) = \frac{\frac{\alpha_1}{\sqrt{2\pi\sigma^2}} exp^{-\frac{\|Mx-y\|_2^2}{2\sigma^2}}}{\alpha_0} \tag{4.65}$$

If we now consider the logarithm of this likelihood, and get rid of all the constants but $\frac{1}{2}$, the log-likelihood maximization problem amounts to:

$$\max_{x\in\mathbb{R}^k} \quad -\frac{1}{2}||Mx-y||_2^2 \tag{4.66}$$

$$\Leftrightarrow \quad \min_{x\in\mathbb{R}^k} \quad \frac{1}{2}||Mx-y||_2^2 \tag{4.67}$$

4.3.3 Weighted Least Square for heteroscedastic data

4.3.3.1 Limits of the least square model for tomography

The Bayesian framework developed in the previous section provides an interesting tool to study the solution of the linear least square. For instance, in some cases, the solution of the

least square may not be satisfying in part because of the discrepancy between the underlying statistical model, and the implicit homoscedatic gaussian assumption of the least square. Here are some of the discrepancies that should be considered:

- The noise or measurment error may not follow a centered gaussian distribution
- The statistics of the vector y may not be homoscedastic.

It appears that both cases are actually relevant for tomographic reconstruction. As stated in section 2.1.2, the vector y is the outcome of multiple stochastic processes, mainly the X-Ray photon statistics that follows a poisson law, and the detector readout noise, which is often modeled as an additive gaussian noise.

The fact that both Poisson, and Gaussian distributions considered for the tomographic model are centered is a good point because expectation and maximum likelihood are equals in both cases, we already exploited this property in order to simplify our tomographic reconstruction model. However, due to the physical nature of photon statistics, that depends on the imaged object itself, we have also seen that each pixel value is a random variable that has its own variance. In this case, one cannot expect for the implicit homoscedastic model performs well for real tomographic experiments.

4.3.3.2 Generalizing least square

When looking at the Bayesian framework exposed in the section 4.3.2, one can notice that the gaussian assumption can be modified in favor of a truly multidimensional heteroscedastic model, assuming that the underlying multivariate gaussian parameters are known in advance. Using the Central-limit theorem, or some apriori on the physical system, on can even extend the gaussian model to approximate more complex statistical distributions with known variance.

Assuming that we know the expectation y, and the covariance matrix of our dataset Σ, we recall that the loglikelihood of a point Mx where $Mx \in \mathbb{R}^n$ can be computed using the following formula

$$L(y) = \frac{1}{\sqrt{(2\pi)^n det(\Sigma)}} e^{-\frac{1}{2}(Mx-y)^\intercal \Sigma^{-1}(Mx-y)} \tag{4.68}$$

$$\tag{4.69}$$

As a covariance matrix, is, by definition symmetric and positive semi definite, it can be diagonalized in an orthogonal basis. In our case, we will focus on positive definite covariance matrices, so that the multidimensional gaussian probability distribution function expressed earlier is valid.

In this framework, the covariance matrix can be written $\Sigma = Q^t D Q$, such that its inverse reads $Q^t D^{-1} Q$, and the following maximum likelihood problem can be casted:

$$\begin{aligned}
& \max_{x \in \mathbb{R}^k} && -\frac{1}{2}(Mx-y)^t \Sigma^{-1}(Mx-y) && (4.70)\\
\Leftrightarrow\ & \min_{x \in \mathbb{R}^k} && \frac{1}{2}(Mx-y)^t Q^t D^{-1} Q(Mx-y) && (4.71)\\
\Leftrightarrow\ & \min_{x \in \mathbb{R}^k} && \frac{1}{2}(Mx-y)^t Q^t D^{-1/2} D^{-1/2} Q(Mx-y) && (4.72)\\
\Leftrightarrow\ & \min_{x \in \mathbb{R}^k} && \frac{1}{2}\|D^{-1/2} QMx - D^{-1/2} Qy\|_2^2 && (4.73)\\
\Leftrightarrow\ & \min_{x \in \mathbb{R}^k} && \frac{1}{2}\|M'x - y'\|_2^2 && (4.74)
\end{aligned}$$

Where $M' = D^{-1/2} QM$ and $y' = D^{-1/2} Qy$

The heteroscedastic case can be treated just like another least square problem, so that all following developments will be valid. In the practical cases, due to the lack of informations regarding the dataset covariance, we will generally assume that Σ is diagonal, and each element will denote the estimated pixel variance. If a poisson/gaussian mixture estimator is available, one can derive an estimate of the actual pixel variance after the Beer-Lambert transformation, otherwise, one has to guess which component dominates the variance, and choose between the homo or heteroscedastic model. A comprehensive study on variance weighted methods has been conducted in [**zeng:16:nww**].

4.3.4 Smoothness and convexity of least square

In the following developments, we will assume that the projector P - backprojector B pair verifies the positive spectral condition studied in [**zeng2000unmatched**], this ensure that the BP operator is positive definite. Although we mainly focused in the chapter 2 and 3 on the quality of the projection operator from a signal processing point of view, we will recall here some of the results that highlight the fact that the properties of the BP matrix are of critical importance in the fromulation of the least square problem.

4.3.4.1 Smoothness

We recall that a l-Lipschitz smooth differentiable function $f : \mathbb{R}^k \to \mathbb{R}$, has its gradient lipshitz-continuous:

$$\|\nabla f(x) - \nabla f(y)\| \leq l\|x-y\|, \forall (x,y) \in \mathbb{R}^k \times \mathbb{R}^k \quad (4.75)$$

If f is a quadratic functional, like the least square, it is easy to show that this amounts to a condition on the operator norm of the Hessian H: $\|H\| \leq l$. The more $\|H\|$ is small, the more the functional is smooth, so that one would usually wish for a small maximum eigenvalue for H in order to get a functional as smooth as possible. Checking this property for an unmatched pair of projector-backprojector can be done easily using the power method.

4.3.4.2 Convexity

The f function defined earlier is convex if the following condition holds:

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y), \forall (x,y) \in \mathbb{R}^k \times \mathbb{R}^k, \lambda \in [0,1] \tag{4.76}$$

Assuming $g(\lambda) = f(\lambda x + (1-\lambda)y)$

When f is differentiable in every point y, there is an equivalent definition: the hyperplan defined by $\nabla f(y), x - y\rangle = 0$ actually defines a supporting hyperplan of the graph of f, and it allows us to define a linear lower bound for f:

$$f(x) \geq f(y) + \langle \nabla f(y), x - y\rangle, \forall (x,y) \in \mathbb{R}^k \times \mathbb{R}^k \tag{4.77}$$

When f is twice differentiable, there is another equivalent definition that follows from Taylor expansion (see [**boyd2004convex**]), based on the positive definiteness of the Hessian of f : H:

$$H \succeq 0 \tag{4.78}$$

In our linear least square, this amounts to the condition derived in [**zeng2000unmatched**] stated earlier. One would usually wish for a large minimum eigenvalue for H in order to get a functional as convex as possible.

4.3.4.3 Strong convexity

For f a differentiable function (but this property can be extended to other functions with the subgradient), we say that f is α strongly convex if:

$$f(x) - f(y) \leq \langle \nabla f(x), x - y\rangle - \frac{\alpha}{2}\|x-y\|_2^2, \forall (x,y) \in \mathbb{R}^k \times \mathbb{R}^k \tag{4.79}$$

$$f(y) \geq f(x) + \langle \nabla f(x), y - x\rangle + \frac{\alpha}{2}\|x-y\|_2^2, \forall (x,y) \in \mathbb{R}^k \times \mathbb{R}^k \tag{4.80}$$

It can be seen on the second line that this property imply that for any strongly convex function, at any point $x \in \mathbb{R}^k$, we can find a quadratic lower bound of f. It can also be shown that the Hessian of a strongly convex function must satisfy $H - \alpha Id \succeq 0$.

Again, we see that in the linear least square relies on the property of BP, and is also linked to the smallest eigen value of the BP matrix. Unfortunately, in practice, retrieving this value with the power method, as presented in [**zeng2000unmatched**] is extremely challenging, because the power method convergence rate depends on the ratio between the norm of the eigenvalues, and that those values tends to be clustered close to zero for ill conditioned systems.

4.4 Krylov based methods

An excellent introduction to Krylov methods, and their role in high performance computing can be found in [**magoulesparallelization**]. In this section, we will try to recall some of the most important features of this algorithm, and how they can help in analyse what tomographic reconstruction.

In this section, we will assume that we want to solve a square problem of the form $M^tMx - M^ty = 0$, which describes the vanishining point of the derivative of the least square.

4.4.1 Taylor expansion: from scalar to matrices

Taylor expansion can easily be applied over differentiable scalar functions of one variables, this leads for instance to the following expansion, for x close to 0:

$$\frac{1}{1-x} = 1 + x + x^2 + x^3 + \dots \tag{4.81}$$

$$= \sum_{n=0}^{\infty} x^n \tag{4.82}$$

It is interesting to notice that the right part of this equality can also be interpreted as the sum of all terms of a geometric series with a common ratio of x, which is valid when $|x| < 1$.

For the sake of our argument, we will assume that, M can be diagonalized in an orthogonal eigenspace (symmetry), so that all its power can be simultaneously diagonalised in the same eigen space, hence provinding an easy way to extrapolate from the scalar case to each diagonal elements. The previous expansion extended to the symmetric matrices case over a real field then reads:

$$(Id - M)^{-1} = Id + M + M^2 + M^3 + \dots \tag{4.83}$$
$$= \sum_{n=0}^{\infty} M^n \tag{4.84}$$

This expansion is valid for matrices whose operator norm $\|M\| = |\lambda(M)_{max}| < 1$, and of course apply for nilpotent matrices, which have all their eigenvalues identically equal to zero, their characteristic polynomial being x^n.

Using $|\lambda(M)_{max}| = c \neq 0$, excluding the already handled case of nilpotent matrices, we can extend the previous power series to all symmetric matrices M:

$$\left(Id - \left(\frac{M}{c}\right)\right)^{-1} = Id + \frac{M}{c} + \left(\frac{M}{c}\right)^2 + \left(\frac{M}{c}\right)^3 + \dots \tag{4.85}$$
$$(cId - M)^{-1} = \frac{1}{c}\left(Id + \frac{M}{c} + \left(\frac{M}{c}\right)^2 + \left(\frac{M}{c}\right)^3 + \dots\right) \tag{4.86}$$
$$= \frac{1}{c}\sum_{n=0}^{\infty}\left(\frac{M}{c}\right)^n \tag{4.87}$$

Withou loss of generality, a variable change $M = cId - P$, where P is also symmetric, gives us

$$P^{-1} = \frac{1}{c}\sum_{n=0}^{\infty}\left(\frac{cId - P}{c}\right)^n \tag{4.88}$$

We will see how this Taylor expansion is linked to preconditionned iterations in the next section

4.4.2 Preconditionned iterations

The approach seen in the section 4.2.7.2, with a-priori decomposition of the matrix M that lead to a fixed point iterations can also be generalized as follows:

$$Mx - y = 0 \tag{4.89}$$
$$(P - (P - M))x) - y = 0 \tag{4.90}$$
$$Px = (P - M)x + y \tag{4.91}$$
$$P^{-1}Px = P^{-1}(P - M)x + P^{-1}y \tag{4.92}$$
$$x = x + P^{-1}(y - Mx) \tag{4.93}$$

This problem can simply be recasted as a fixed point search problem, that can be solved using the following iterations:

$$x^{k+1} = x^k + P^{-1}(y - Mx^k) \tag{4.94}$$
$$x^{k+1} = x^k + P^{-1}r^k \tag{4.95}$$

Where $r^k = y - Mx^k$ is called the residual and P is called the preconditionner.

The preconditionner could feature some useful properties, like the fact that P^{-1} is easy to compute, and eventually P has a low condition number, or P may even carry some a-priori informations over the problem. In the case of tomography, researcher have experienced interesting convergence speedup from using the filtering stage of the filtered backprojection, see the analysis of the iterative FDK and iterative FBP from [**mory2014tomographie**] and [**zeng2000unmatched**].

We can notice that, if we choose the preconditionner such that $P^{-1} = \frac{1}{\lambda(M^\intercal M)_{max}} M^\intercal$ we get an instance of the gradient descent algorithm.

4.4.3 Krylov subspaces

Let reconsider the equation of preconditionned iterations, where we will incorporate the preconditionner P directly to M and y such that we have our new M equivalent to $P^{-1}M$ and our new y equivalent to $P^{-1}y$. The preconditionned iterations now reads

$$\vec{x^{k+1}} = (Id - M)\vec{x^k} + y \tag{4.96}$$

assuming $\vec{x^0} = y$, we can write

$$\vec{x^0} = y \tag{4.97}$$
$$\vec{x^1} = (Id - M)y + y \tag{4.98}$$
$$\vec{x^2} = (Id - M)^2 y + (Id - M)y + y \tag{4.99}$$
$$\tag{4.100}$$

Now, using the matrix version of the Taylor expansion exposed in the previous section, we can give another proof of the convergence of the x^k series:

$$\vec{x^k} = \sum_{n=0}^{k} (Id - M)^n y \tag{4.101}$$

$$\lim_{k\to\infty} \vec{x^k} = (Id - (Id - M))^{-1} y \tag{4.102}$$

$$\lim_{k\to\infty} \vec{x^k} = M^{-1} y \tag{4.103}$$

$$\tag{4.104}$$

The fact that the inverse of M in y can be expressed in a basis made of vectors of the form $M^n y, n \in 0, 1, \ldots$ was first discovered by Krylov, using a more general approach based on the Cayley-Hamilton theorem, which do not impose condition over the spectral radius, and where the polynomial in M was the characteristic polynomial of the matrix M.

In the general case, the space $\mathcal{K}_r$ spanned by the r first vectors : $M^n y, n \in 0, 1, \ldots, r-1$ is known as the order-r Krylov subspace, and it can be shown that its basis, made of the vectors $M^n y, n \in 0, 1, \ldots, r-1$ is free while $r \leq r_{max}$, with $r_{max} < k$

4.4.4 Practical considerations on krylov subspaces

From the previous section presenting a polynomial expression in M^n, on can think about deriving the Krylov matrix K:

$$\begin{pmatrix} y & My & M^2 y & \ldots & M^{n-1} y \end{pmatrix} \tag{4.105}$$

Unfortunately, in many cases, this basis cannot be used as is to express the solution for numerical reasons: assuming M can be diagonalized and has n different eigenvalues $\lambda(M)_0, \lambda(M)_1, \ldots \lambda(M)_{n-1}$ associated with the eigenvectors $\vec{v_0}, \vec{v_1}, \ldots \vec{v_{n-1}}$ we can write

$$y = \sum_{i=0}^{n-1} \alpha_i \vec{v_i} \tag{4.106}$$

$$M^n y = \sum_{i=0}^{n-1} \alpha_i \lambda(M)_i^n \vec{v_i} \tag{4.107}$$

When the ratio $\left(\frac{\alpha_{max}}{\alpha_i}\right)\left(\frac{\lambda(M)_{max}}{\lambda(M)_i}\right)^n$ where max stands for the index of the maximum eigenvalue, exceeds 2^n, the accuracy of any krylov subspace method would collapse. n being the number of significand bits in the floating point representation of a computer.

To overcome this problem, Arnoldi used the simple principle or Gram-Schmidt orthogonalization process for each new multiple of My in order, not only to obtain a basis of the

Krylov space, but an orthogonal basis. Later, Lanczos found out that, in the case of symmetric matrices, the upper Hessenberg matrix used for the Gram-Schmidt process of Arnoldi method was actually also symmetric, hence leading to a tridiagonal matrix. This observation has a huge impact on algorithm complexity, because it offers a method to build iteratively an orthonormal basis where each new vector is orthogonalized using a fixed number of steps (actually 2 orthogonalization and one normalization).

Lanczos method can be seen as a matrix factorization method, that has two levels of decomposition. The first level is basically the tridiagonal Gram-Schmidt process that yield $V_p H_{pp} V_p^\intercal = M$, with V_p a unitary matrix. The second level allows to compute the diagonalization of H_{pp} as a Choleski decomposition of the form $L_p D_p L_p^\intercal$. Although there does not seem to be much litterature about the use of Lanczos method for computing eigenvalues of tomographic systems more efficiently than with the power method, this topic goes beyond the scope of our work.

4.4.5 Krylov basis and practical resolution of linear equalities

As seen earlier, the solution $M^{-1}y$ of $Mx = y$ can be expressed in terms of $M^p y, p = 0, 1, \ldots, r-1$, which led us to define the order-r Krylov basis.

We have then seen that Arnoldi and Lanczos provided a way to express the p-order Krylov basis V_p in a numerically stable way . We can now define $\vec{x_p}$, the approximation of the solution of $Mx - y = 0$ in the order-p Arnoldi basis:

$$\vec{x_p} = V_p \vec{s_p} \tag{4.108}$$

From this can be defined common error metrics over this approximate solution, using :

- The error vector: $\vec{e_p} = x - \vec{x_p}$
- The residual vector: $\vec{r_p} = y - M\vec{x_p}$

Krylov subspace related methods aims at minimizing $\vec{e_p}$ or $\vec{r_p}$ under appropriate norms. One of the most reknown method, first derived by Lanczos aims at minimizing $Lcz(\vec{x_p})$ the scalar product of those two error vector, that can be also interpreted as the squared norm of the error vector using the scalar product defined by the matrix M:

$$
\begin{aligned}
Lcz(\vec{x_p}) &= \|\vec{e_p}\|_M^2 && (4.109)\\
&= \|x - \vec{x_p}\|_M^2 && (4.110)\\
&= (x - \vec{x_p})^\intercal M(x - \vec{x_p}) && (4.111)\\
&= \vec{e_p}^\intercal.(Mx - M\vec{x_p}) && (4.112)\\
&= \vec{e_p}^\intercal.(y - M\vec{x_p}) && (4.113)\\
&= \vec{e_p}^\intercal.\vec{r_p} && (4.114)\\
& && (4.115)
\end{aligned}
$$

The Lanczos method can be viewed as the task of finding the vector $\vec{x_p} \in \mathcal{K}_p$ such that its distance to x is minimum in the inner product space $\mathcal{M}^n$ defined by M from $\mathbb{R}^n$, for this method, we need M to be symmetric and positive definite. The vector $\vec{x_p}$ that satisfies this definition is the orthogonal projection of x over $\mathcal{K}_p$ using the inner product $\langle ., . \rangle_M$. Following the definition of the inner product $\langle ., . \rangle_M$ and the definition of the orthogonal projection operator,

$$
\begin{aligned}
P_{\mathcal{K}_p} : \mathcal{M}^n &\to \mathcal{K}_p && (4.116)\\
x &\to \vec{x_p} && (4.117)
\end{aligned}
$$

in the space $\mathcal{M}^n$ we have that the range $\mathcal{K}_p$ and the nullspace $\mathcal{M}^n \backslash \mathcal{K}_p$ are orthogonal subspaces in direct sum, which means for us that $\mathcal{M}^n = \mathcal{K}_p \bigoplus_M \mathcal{M}^n \setminus \mathcal{K}_p$ and :

$$
\forall (a, b) \in \mathcal{M}^n \times \mathcal{M}^n \setminus \mathcal{K}_p, \langle a, b \rangle_M = 0 \tag{4.118}
$$

By construction, it is obvious that $\vec{e_p} = x - \vec{x_p} \in \mathcal{M}^n \setminus \mathcal{K}_p$ so we have:

$$
\begin{aligned}
\langle \vec{x_p}, x - \vec{x_p} \rangle_M = 0 \quad \forall \vec{x_p} \in \mathcal{K}_p, \text{ and } x \in \mathbb{R}^n && (4.119)\\
\vec{x_p} M(x - \vec{x_p}) = 0 && (4.120)\\
\vec{x_p}.(y - M\vec{x_p}) = 0 && (4.121)\\
\vec{x_p}.\vec{r_p} = 0 && (4.122)
\end{aligned}
$$

So we have that, in $\mathbb{R}^n$, the residual $\vec{r_p}$ is always orthogonal to all vectors in $\mathcal{K}_p$ which is a key feature of the conjugate gradient we will describe in the next section.

4.4.6 Conjugate Gradient approach

Although the previous algorithm features some interesting properties, we can notice that it is not exactly a method that iteratively builds a solution vector. Instead, it iteratively builds a new problem, where $\vec{s_p}$, the M-projection of the solution $\hat{x}$ in the Krylov sub-space $\mathcal{K}_p$ expressed in the basis V_p can be solved whithout too much effort, thanks to the triangular/-diagonal structure of the $LDL^\intercal$ Choleski factorization.

An alternative formulation where the current solution $\vec{x_p}$ would be iteratively constructed in the Krylov subspace $\mathcal{K}_p$ using sequential vectors $\vec{v_k}, k = 0, 1, \ldots, p-1$ from the basis V_p

would allow us to save the computation needed to solve the full projection problem at each iteration, and the main update would look like:

$$x_{\vec{p+1}} = \vec{x_p} + \alpha_p \vec{v_p} \tag{4.123}$$

4.4.7 General Idea of Conjugate Gradient in the framework of Krylov methods

We must recall that the solution vector $\vec{s_p}$ at each step is the orthogonal projection of the general solution $\hat{x}$ over the Arnoldi basis V_p. But it is orthogonal with respect to a specific inner product definition that uses $M \in S^n_{++}$, although the Arnoldi basis V_p column vectors are orthogonal with respect to the canonical inner product based on the identity matrix. So there is no guarantee that the p first coordinates of $\vec{s_p}$ in V_p will remain the same in $s_{\vec{p+1}}$ expressed in the new basis V_{p+1}. This is why, at each iteration of the previous algorithm, the linear combination of vectors from V_p had to be fully recomputed by solving a structured linear set of equations.

To overcome this problem, let's first recall that the problem to solve at each iteration as seen previously, reads:

$$H_{pp}\vec{s_p} - V_p^\intercal y = 0 \tag{4.124}$$
$$H_{pp}\vec{s_p} = V_p^\intercal y \tag{4.125}$$
$$H_{pp}\vec{s_p} = \vec{y_p} \tag{4.126}$$

As $\vec{y_p}$ and $y_{\vec{p+1}}$ only differs in the last coordinate that have been added to $y_{\vec{p+1}}$, as seen earlier, if we want to have a coordinate-wise resolution of the problem in the basis V_p, the same property should apply to $H_{pp}\vec{s_p}$, which could be translated in:

$$\begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & 1 & 0 \\ 0 & \dots & 0 & 0 \end{pmatrix} H_{p+1p+1} s_{\vec{p+1}} = H_{pp}\vec{s_p} \tag{4.127}$$

$$\tag{4.128}$$

Which imply that H_{pp} should be constructed iteratively so that H_{pp} and H_{p+1p+1} only differs from the last line and the last column, but also that H_{pp} should be diagonal for every p.

To achieve this simple construction scheme, it is obvious that H_{pp} should be diagonal. To do so, we have to find a new Krylov basis W_p such that our new H_{pp} is diagonal:

$$H_{pp} = W_p^\intercal M W_p \tag{4.129}$$

Fortunately, we don't need to take the Krylov basis design problem from the beginning to handle this constraint. We have already seen that the residue $\vec{r_p}$ is orthogonal to $\mathcal{K}_p$, and by construction, $\vec{r_p} \in \mathcal{K}_{p+1}$:

$$\vec{r_p} = M\vec{x_p} - y \tag{4.130}$$

with $y \in \mathcal{K}_1$, $\vec{x_p} \in \mathcal{K}_p$ so $M\vec{x_p} \in \mathcal{K}_{p+1}$. We conclude that $\vec{r_p}$ lies in the orthogonal complement of $\mathcal{K}_p$ in $\mathcal{K}_{p+1}$ that we can write $\mathcal{K}_{p+1} \setminus \mathcal{K}_p$ which is of dimension 1 and has an orthonormal basis $\vec{v_p}$. So every $\vec{r_p}$ can be written $\alpha\vec{v_p}$.

4.4.8 Construction of a new Krylov space basis

The last remark in the previous part can be directly translated into a matrix equality, let R_p be the matrix of all ordered p first residual vectors $\vec{r_k}, k \in 0, 1, \ldots, p-1$, we can write:

$$R_p = V_p \Delta_p \tag{4.131}$$

Where Δ_p is a diagonal matrix, and we can now extend properties valid for the decomposition of H_{pp} :

$$\begin{aligned}
& R_p^\intercal M R_p && (4.132)\\
=& \Delta_p^\intercal V_p^\intercal M V_p \Delta_p && (4.133)\\
=& \Delta_p^\intercal H_{pp} \Delta_p && (4.134)\\
=& \tilde{H}_{pp} && (4.135)\\
& && (4.136)
\end{aligned}$$

Where the matrix $\tilde{H}_{pp}$ that is equivalent to the projection of the linear operator M over $\mathcal{K}_p$ expressed in the basis R_p is still tridiagonal and positive definite, and admits an alternative Choleski decomposition of the form $\tilde{H}_{pp} = \tilde{L}_p \tilde{D}_p \tilde{L}_p^\intercal$ such that we can write

$$\begin{aligned}
& R_p^\intercal M R_p && = \tilde{L}_p \tilde{D}_p \tilde{L}_p^\intercal && (4.137)\\
\Leftrightarrow & \Delta_p^\intercal V_p^\intercal M V_p \Delta_p && = \tilde{L}_p \tilde{D}_p \tilde{L}_p^\intercal && (4.138)\\
\Leftrightarrow & \tilde{L}_p^{-1} \Delta_p^\intercal V_p^\intercal M V_p \Delta_p \tilde{L}_p^{-\intercal} && = \tilde{D}_p && (4.139)\\
\Leftrightarrow & W_p && = R_p \tilde{L}_p^{-\intercal} && (4.140)
\end{aligned}$$

Column vectors from W_p are derived from residual vectors, they form a basis of $\mathcal{K}_p$ and they features the desired property stated in the previous section: they are all orthogonal to each other with respect to the inner product $\langle ., . \rangle_M$ in $\mathcal{M}^n$.

Knowing that $\tilde{L}_p$ is bidiagonal inferior by construction, as seen in the Choleski factorization, we can also derive a short recurrence pattern over $\vec{w_0}, \vec{w_1}, \ldots, \vec{w_{p-1}}$ the column vector of W_p, assuming that $\gamma_0, \gamma_1, \ldots, \gamma_{p-1}$ are the subdiagonal elements of $\tilde{L}_p$:

$$W_p \tilde{L}_p^t = R_p \tag{4.141}$$

$$\left(\begin{pmatrix} \\ \vec{w_0} \\ \\ \end{pmatrix} \begin{pmatrix} \\ \vec{w_1} \\ \\ \end{pmatrix} \ldots \begin{pmatrix} \\ \vec{w_{p-1}} \\ \\ \end{pmatrix}\right) \begin{pmatrix} 1 & \gamma_0 & 0 & \ldots & 0 \\ 0 & 1 & \gamma_1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 1 & \gamma_{p-2} \\ 0 & \ldots & \ldots & 0 & 1 \end{pmatrix} = \left(\begin{pmatrix} \\ \vec{r_0} \\ \\ \end{pmatrix} \begin{pmatrix} \\ \vec{r_1} \\ \\ \end{pmatrix} \ldots \begin{pmatrix} \\ \vec{r_{p-1}} \\ \\ \end{pmatrix}\right) \tag{4.142}$$

This formulation may give us a way to construct w_{j+1} with a simple iterative scheme, assuming that $\vec{x_0} = \vec{0}$:

$$W_p \tilde{L}_p^\intercal = R_p \tag{4.143}$$
$$\Rightarrow \vec{r_0} = \vec{w_0} = y - M\vec{x_0} \text{ and} \tag{4.144}$$
$$\vec{r_{j+1}} = \vec{w_{j+1}} + \gamma_j \vec{w_j} \tag{4.145}$$
$$\Leftrightarrow \vec{w_{j+1}} = \vec{r_{j+1}} - \gamma_j \vec{w_j} \tag{4.146}$$
$$\tag{4.147}$$

This relationship between $\vec{w_{j+1}}$ and $\vec{w_j}$ can be further exploited, if we use the fact that all vectors from W_p are pairwise orthogonal with respect to the inner product defined by M:

$$\langle \vec{w_{j+1}}, \vec{w_j} \rangle_M = 0 \tag{4.148}$$
$$\Leftrightarrow \langle \vec{r_{j+1}} - \gamma_j \vec{w_j}, \vec{w_j} \rangle_M = 0 \tag{4.149}$$
$$\Leftrightarrow \vec{r_{j+1}}^\intercal M \vec{w_j} - \gamma_j \vec{w_j}^\intercal M \vec{w_j} = 0 \tag{4.150}$$
$$\Leftrightarrow \gamma_j = \frac{\vec{r_{j+1}}^\intercal M \vec{w_j}}{\vec{w_j}^\intercal M \vec{w_j}} \tag{4.151}$$

We now need to see if $\vec{r_{j+1}}$ can be found using a simple recursion. To do so, we have to use the coordinate-wise construction of w_{j+1} exposed previously :

$$\vec{x_{j+1}} = \vec{x_j} + \alpha_j \vec{w_j} \tag{4.152}$$
$$M\vec{x_{j+1}} - y = M\vec{x_j} + \alpha_j M\vec{w_j} - y \tag{4.153}$$
$$\vec{r_{j+1}} = \vec{r_j} + \alpha_j M\vec{w_j} \tag{4.154}$$
$$\tag{4.155}$$

Here, we can use the fact that $\vec{r_p}$ is orthogonal to $\mathcal{K}_p$, and especially $\vec{r_j}$ is orthogonal to $\vec{w_{j-1}}$ so that we can write:

$$\langle \vec{r_{j+1}}, \vec{w_j} \rangle = 0 \tag{4.156}$$
$$\Leftrightarrow \langle \vec{r_j} + \alpha_j M\vec{w_j}, \vec{w_j} \rangle = 0 \tag{4.157}$$
$$\Leftrightarrow \langle \vec{r_j}, \vec{w_j} \rangle + \alpha_j \langle M\vec{w_j}, \vec{w_j} \rangle = 0 \tag{4.158}$$
$$\Leftrightarrow \vec{r_j}^\intercal \vec{w_j} + \alpha_j \vec{w_j}^\intercal M\vec{w_j} = 0 \tag{4.159}$$
$$\Leftrightarrow \alpha_j = \frac{-\vec{r_j}^\intercal \vec{w_j}}{\vec{w_j}^\intercal M\vec{w_j}} \tag{4.160}$$

4.4.9 Conjugate Gradient algorithm in practice

The force of the Conjugate gradient algorithm lies in the fact that it iteratively builds a diagonal problem which can be solved one coordinate per iteration, without "forgetting" about the previous iterations. More interestingly, it is not solved in the canonical coordinate system of $\mathbb{R}^n$ but instead in a basis of pairwise orthogonal vectors with respect to the matrix M, called conjugate vectors, which are iteratively built using the gradient of the function $f(x)$ we want to minimize at the current solution estimate : $f(x) = \frac{1}{2}(x - \vec{\hat{x}})(Mx - y)$.

By design, these vectors are able to catch a large part of the error at every iteration, and especially, if the matrix M defines an ellipsoid with an important anisotropy, the conjugate strategy helps to avoid the drawback of descent methods that are often “stuck” in valleys.

The algorithm reads:

4.4.10 Going further with the Krylov methods

Although conjugate gradient is one of the most simple Krylov method, which makes it a good choice for large scale optimization, it does not enjoy the nice property of monotically decreasing residual cost, because it is designed to reduce de $Lcz()$ cost function seen earlier, however, it is generally extremely fast in practice when the problem is not too ill conditionned. There are other Krylov methods, based on other objective functions, such as GMRES, Biconjugate gradient, or SYMMLQ. GMRES for instance can handle non symmetric matrices, which could

Algorithm 1 Solve $Mx - y = 0$

Require: $M \in S_{++}^n$ the positive definite cone
  Initialization
  $\vec{r_0} = y$
  $\vec{w_0} = \vec{r_0}$
  $j = 1$
  Iterations
  while $j \leq n$ **do**
    $\alpha_{i-1} = \frac{\vec{r_{i-1}}^\intercal \vec{w_{i-1}}}{\vec{w_{i-1}}^\intercal M \vec{w_{i-1}}}$
    $\vec{x_i} = \vec{x_{i-1}} + \alpha_{i-1} \vec{w_{j-1}}$
    $\vec{r_i} = \vec{r_{i-1}} - \alpha_{i-1} M \vec{w_{j-1}}$
    if $\vec{r_i} \neq \vec{0}$ **then**
      $\gamma_{i-1} = \frac{\vec{r_i}^\intercal M \vec{w_{i-1}}}{\vec{w_{i-1}}^\intercal M \vec{w_{i-1}}}$
      $\vec{w_i} = \vec{r_i} - \gamma_{i-1} \vec{w_{j-1}}$
    else
      End iterations
    end if
  end while

be interesting in tomography when the product of forward and backward operators yield a matrix that differs too much from a symmetric matrix, see the work in [**coban2014regularised**] where the author instanciated the GMRES algorithm for X-Ray CT reconstruction.

4.5 Gradient descent

4.5.1 A low cost second order method

As seen in 4.4, there are multiple ways to interpret the gradient descent algorithm. An interesting interpretation consist in looking at gradient descent as a second order optimization method, with a poor estimate of the Hessian. To understand this concept, let's first take a look at how most of the second order methods have been derived:

4.5.1.1 Newton method

Newton method is a really simple method originally designed in order to find iteratively the root of a function (where it is zero valued), we will see later some conditions that must be met in order to ensure the algorithm convergence.

One of the first condition is simply to be sure that the function under consideration, for instance $f : \underset{\mathbb{R}^n}{x} \mapsto \underset{\mathbb{R}}{f(x)}$, crosses the X axis of the graph at some point.

The basic idea of newton method is to linearize the function at the current estimate, find the root of the linear approximation, and assign its value to the current estimate before starting a new iteration.

The linear estimation f_{lin} at point a, using Taylor serie gives:

$$f_{lin,a}(x) = f(a) + \langle x - a, \nabla f(a) \rangle \tag{4.161}$$

Where we recall that $\nabla f(a) = \begin{pmatrix} \frac{\partial f(a)}{\partial x_0} \\ \frac{\partial f(a)}{\partial x_1} \\ \vdots \\ \frac{\partial f(a)}{\partial x_{n-1}} \end{pmatrix}$

Now, equating $f_{lin,a}(x)$ to zero leads to:

$$f_{lin,a}(x) = 0 \tag{4.162}$$
$$f(a) + \langle \nabla f(a), x - a \rangle = 0 \tag{4.163}$$
$$\langle \nabla f(a), x \rangle - \langle \nabla f(a), a \rangle = -f(a) \tag{4.164}$$
$$\langle \nabla f(a), x \rangle = \langle \nabla f(a), a \rangle - f(a) \tag{4.165}$$

This simple linear equality defines an hyperplan onto which the current solution can be projected in order to get the next iterate.

Here is a simple overview of the process for a 1D case, see figure 4.5:

4.5.1.2 Extending Newton method for convex optimization

We are generally not really interested in finding the root of an objective function, plus, there is very few chances that we would ever be able to prove it exists. However, when manipulating functions that are convex and twice differentiable, we know, thanks to Fermat theorem, that its derivative vanishes at the optimal point. So finding the root of the derivative of our objective seems to be a much more interesting challenge.

We will replace $f(x)$ by $\nabla f(x)$ in our previous equations to obtain the desired equation update:

$$\nabla f_{lin,a}(x) = 0 \tag{4.166}$$
$$\nabla f(a) + H_f(a), (x - a) = 0 \tag{4.167}$$

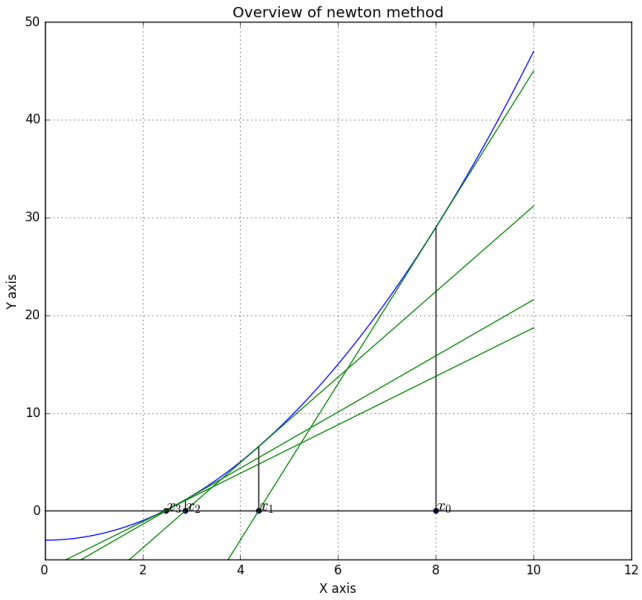


Figure 4.5: Simple instance of Newton algorithm for a 1D quadratic function

Here we recall that $H_f(a)$ the Hessian of f in a can be defined as

$$H_f(a) = \nabla\nabla f(a) = \begin{pmatrix} \frac{\partial f(a)}{\partial x_0 \partial x_0} & \frac{\partial f(a)}{\partial x_1 \partial x_0} & \cdots & \frac{\partial f(a)}{\partial x_{n-1} \partial x_0} \\ \frac{\partial f(a)}{\partial x_0 \partial x_1} & \frac{\partial f(a)}{\partial x_1 \partial x_1} & \cdots & \frac{\partial f(a)}{\partial x_{n-1} \partial x_1} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f(a)}{\partial x_0 \partial x_{n-1}} & \frac{\partial f(a)}{\partial x_1 \partial x_{n-1}} & \cdots & \frac{\partial f(a)}{\partial x_{n-1} \partial x_{n-1}} \end{pmatrix} \tag{4.168}$$

The equation becomes:

$$\nabla f(a) + H_f(a)x - H_f(a)a = 0 \tag{4.169}$$
$$H_f(a)x = H_f(a)a - \nabla f(a) \tag{4.170}$$
$$x = H_f^{-1}(H_f(a)a - \nabla f(a)) \tag{4.171}$$
$$x = a - H_f^{-1}\nabla f(a) \tag{4.172}$$

We can notice that we have to compute H_f^{-1}, the inverse of H_f. Although H_f is a symmetric matrix, this should be considered as a very difficult task in the general case, especially when dealing with high dimensional problems.

4.5.1.3 Second order method in the linear case

For a simple quadratic problem such as the least square minimization, it is very simple to prove that the Newton method converges in only one step, and is equivalent to compute the pseudo inverse. Let's define the following problem:

- $f(x) = \|Mx - y\|_2^2$
- $\nabla f(x) = M^t Mx - M^t y$
- $H_f(x) = M^t M$

The newton method can be used on the gradient, because the probem is convex, and twice differentiable. Assuming an initial solution vector x_0, the first iterate reads:

$$\begin{aligned} x_1 &= x_0 - H_f^{-1}\nabla f(x_0) && (4.173)\\ &= x_0 - (M^t M)^{-1}(M^t M x_0 - M^t y) && (4.174)\\ &= x_0 - (M^t M)^{-1} M^t M x_0 + (M^t M)^{-1} M^t y && (4.175)\\ &= (M^t M)^{-1} M^t y && (4.176)\\ &= M^+ && (4.177) \end{aligned}$$

4.5.1.4 Low cost Hessian estimation

Newton method is then useless for our least square estimate problem, because it amounts to the pseudo inverse, however, it is interesting to notice that one can use a "low cost" estimation of H_f^{-1} instead of computing the full inverse. Let's take a closer look at the algorithm update:

$$x^{k+1} = x^k - H_f^{-1}\nabla f(x^k) \tag{4.178}$$

We can reformulate this update as a fixed point method:

$$\begin{aligned} x^k &= x^k - H_f^{-1}\nabla f(x^k) && (4.179)\\ x^k &= Tx^k && (4.180) \end{aligned}$$

We recall that, from the Picard fixed point theorem, this class of algorithm converges if the operator $T : x \to x - H_f^{-1}\nabla f(x)$ is a strict contraction. In our least square case, this reduces to an operator norm condition:

The method converges if it exists $\rho \in [0,1[$ such that

$$\forall (x,x') \in \mathbb{R}^n \times \mathbb{R}^n, x \neq x' : \quad \|Tx - Tx'\| \leq \rho \|x - x'\| \tag{4.181}$$
$$\|x - H_f^{-1}\nabla f(x) - x' + H_f^{-1}\nabla f(x')\| \leq \rho \|x - x'\| \tag{4.182}$$
$$\|x - x' + H_f^{-1}\nabla f(x' - x)\| \leq \rho \|x - x'\| \tag{4.183}$$
$$\|(Id - H_f^{-1}\nabla f)(x - x')\| \leq \rho \|x - x'\| \tag{4.184}$$
$$\|(Id - H_f^{-1}\nabla f)\|\|(x - x')\| \leq \rho \|x - x'\| \tag{4.185}$$
$$\|Id - H_f^{-1}\nabla f\| \leq \rho \tag{4.186}$$

An extremely simple approximation $A_{H_f^{-1}}$ of H_f^{-1} that is easily invertible, and satisfies the above condition is $A_{H_f^{-1}} = \frac{1}{\|H_f\|} Id$. This method is a simple instance of the gradient descent, but it must be noticed that, although it ensure convergence from any starting point, taking the inverse of the upper bound of the Hessian norm has a great chance to be an understimated value of the optimal step size in many cases. The pathologic cases includes the ill conditionned matrices, that tends to exhibit large anisotropy in the corresponding ellipsoid, resulting in deep "valleys" in the function graph, where the descent step size is usually very short to ensure monotonnic convergence.

4.5.1.5 Secant equation and quasi Newton method

The fact that inverting H_f is a challenging task, gave rise to a variety of optimization methods, where H_f^{-1} or $H_f^{-1}\nabla f(x^k)$ is approximated. Those method are known as quasi-Newton methods.

In the general case, we recall that the second order Taylor expansion of a function f as defined above reads

$$f(x^k + \Delta x) \approx f(x^k) + \langle \nabla f(x^k), \Delta x \rangle + \frac{1}{2} \Delta x^t H_f \Delta x \tag{4.187}$$

Which derivative with respect to Δx gives

$$\nabla f(x^k + \Delta x) \approx \nabla f(x^k) + H_f \Delta x \tag{4.188}$$

When looking for a Δx such that this expression vanishes, we have the Quasi Newton relationship or secant equation as seen previously:

$$0 = \nabla f(x^k) + H_f \Delta x \tag{4.189}$$
$$H_f \Delta x = -\nabla f(x^k) \tag{4.190}$$
$$\Delta x = -H_f^{-1} \nabla f(x^k) \tag{4.191}$$

4.5.1.6 Barzilai-Borwein and quasi Newton method

In 1988, in "Two-Point Steap Size Gradient Methods" [**barzilai1988two**], Barzilai and Borwein derived a two point approximation for the secant equation seen earlier that reads

$$H_f(x^{k+1} - x^k) = \nabla f(x^k) - \nabla f(x^{k-1}) \tag{4.192}$$
$$x^{k+1} - x^k = -H_f^{-1}(\nabla f(x^k) - \nabla f(x^{k-1})) \tag{4.193}$$
$$\Delta x = -H_f^{-1} \Delta \nabla f(x) \tag{4.194}$$

Where $\Delta x = x^{k+1} - x^k$ and $\Delta \nabla f(x) = \nabla f(x^k) - \nabla f(x^{k-1})$.

Barzilai-Borwein idea is simply to assume that H_f^{-1} writes λId, like in the gradient descent method, and then looks for the λ that is close to satisfy their two point Quasi Newton relationship in the least square sense:

$$\lambda^\star = \underset{\lambda \in \mathbb{R}}{argmin} \quad \|\Delta x + \lambda Id \Delta \nabla f(x)\|_2^2 \tag{4.195}$$

The minimum of this simple one dimensional objective is attained in:

$$\lambda^\star = -\frac{\langle \Delta \nabla f(x), \Delta x \rangle}{\langle \Delta \nabla f(x), \Delta \nabla f(x) \rangle} \tag{4.196}$$

The newton method updates, that previously read

$$x^{k+1} = x^k + \Delta x \tag{4.197}$$
$$x^{k+1} = x^k - H_f^{-1} \nabla f(x^k) \tag{4.198}$$

Now reads

$$x^{k+1} = x^k - \frac{\langle \Delta \nabla f(x), \Delta x \rangle}{\langle \Delta \nabla f(x), \Delta \nabla f(x) \rangle} \nabla f(x^k) \tag{4.199}$$

It can be noticed that the equation 4.195, can also be rewritten, symmetrically with

$$\lambda^\star = \underset{\lambda \in \mathbb{R}}{argmin} \quad \|\lambda \Delta x + \Delta \nabla f(x)\|_2^2 \tag{4.200}$$

And another algorithm can be drawn from there with

$$\lambda^\star = -\frac{\langle \Delta x, \Delta x \rangle}{\langle \Delta x, \Delta \nabla f(x) \rangle} \tag{4.201}$$

But then the interpretation in this cas is less intuitive.

4.5.1.7 Going further with Barzilai-Borwein acceleration

Although it result in a non-monotonic algorithm, the BB[1] is still a good choice for gradient descent step size regarding other approaches like the classical Lipschitz coefficient based step size, which has been reported to provide a slow convergence speed, see [**wright2009sparse**]. Computing the BB step could also be considered as a fast operation, because it only requires one dot product, and one $\|\cdot\|_2^2$ norm operator, which are reduction operation that can be carried out efficiently in parallel. It is even less costly than the conjugate gradient method exposed in 4.4, because it potentially allows to save the computation of one or two matrix-vector product (depending on wether the forward and backward operators are adjoints of each other).

In 2009, Wright, Nowak and Figueiredo published the Sparse Reconstruction by Separable Approximation method (SpaRSA, see [**wright2009sparse**]) which enjoyed a tremendous success from the machine learning and signal processing community. One of the instance of this algorithm used the BB strategy to derive a non monotic but fast algorithm to perform sparse regression.

4.5.2 Cauchy step size or the steepest descent

If we take a look back to the design of the BB method, we can see that the authors have designed a step-size according to a cost function exposed in 4.195. Although we won't provide the proof here, Cauchy used a similar method, but with a more intuitive cost function, that is chosen to ensure monotonic convergence at every step, but in an adaptive fashion:

$$\lambda^k = \underset{\lambda\in\mathbb{R}}{argmin} \quad f(x^k - \lambda\nabla f(x^k)) \tag{4.202}$$

This simple one dimensional objective being differentiable and convex, a minimum can easily be found for the point where the gradient vanishes. Hopefully there is a closed form solution for the least square formulation $f(x) = \|Mx-y\|_2^2$, whose gradient reads $\nabla f(x^k) = M^tMx^k - M^ty$:

$$\begin{aligned} c(\lambda) &= \|M(x^k - \lambda\nabla f(x^k)) - y\|_2^2 && (4.203)\\ &= (x^k - \lambda\nabla f(x^k))^tM^tM(x^k - \lambda\nabla f(x^k)) + y^ty - 2yM(x^k - \lambda\nabla f(x^k)) && (4.204)\\ &= \lambda^2(\nabla f(x^k))^tM^tM(\nabla f(x^k)) \\ &\quad - 2\lambda(\nabla f(x^k))^tM^tMx^k + 2\lambda yM\nabla f(x^k) && (4.205)\\ &\quad + (x^k)^tM^tMx^k + y^ty - 2yMx^k \end{aligned}$$

[1] *Barzilai Borwein*

This simple second order polynomial can be differentiated, and equated to zero:

$$
\begin{aligned}
0 &= \frac{\partial c(\lambda)}{\partial \lambda} && (4.206)\\
&= 2\lambda(\nabla f(x^k))^t M^t M(\nabla f(x^k)) - 2(\nabla f(x^k))^t M^t M x^k + 2yM\nabla f(x^k) && (4.207)\\
\Leftrightarrow \lambda &= \frac{(\nabla f(x^k))^t M^t M x^k - y^t M \nabla f(x^k)}{(\nabla f(x^k))^t M^t M(\nabla f(x^k)} && (4.208)\\
&= \frac{(M^t M x^k - M^t y)^t M^t M x^k - y^t M(M^t M x^k - M^t y)}{(\nabla f(x^k))^t M^t M(\nabla f(x^k)} && (4.209)\\
&= \frac{(M^t M x^k)^t (M^t M x^k) - 2yM(M^t M x^k) + y^t M M^t y)}{(\nabla f(x^k))^t M^t M(\nabla f(x^k)} && (4.210)\\
&= \frac{(M^t M x^k - M^t y)^t (M^t M x^k - M^t y)}{(\nabla f(x^k))^t M^t M(\nabla f(x^k)} && (4.211)\\
& && (4.212)
\end{aligned}
$$

This last expression of λ, evaluated for each new iterate k then reads:

$$\lambda^k = \frac{\|\nabla f(x^k)\|_2^2}{\|M\nabla f(x^k)\|_2^2} \tag{4.213}$$

This solution is also interesting in the sense it chooses the optimal step-size at each iterate of the gradient descent, in the least square metric. This way, one can evaluate the best "local" step size at each iteration, with a simple calculation: a few matrix-vector products and dot products at each step. Technically this solution should offer a better convergence speed than the lipschitz-based step-size, which is computed only once, while providing a monotonic decrease of the cost function, that Barzilai-Borwein step-size cannot ensure.

4.5.3 Gradient descent as a proximity operator

Another framework can be used to provide an interesting interpretation of the gradient step size : proximal mappings. Indeed, a single step of gradient descent for the objective function f can be viewed as a proximal mapping for the objective g_{x^k}, the first order taylor development of f in x^k, as seen in 4.5.1.1:

$$g_{x^k}(x) = f(x^k) + \langle x - x^k, \nabla f(x^k)\rangle \tag{4.214}$$

Let's check that the proximity operator of g_{x^k} in x^k amounts to a gradient descent step:

$$prox_{\lambda g_{x^k}}(x^k) = \underset{x\in\mathbb{R}^n}{argmin} \quad \underbrace{f(x^k)}_{\text{constant}} + \underbrace{\langle x - x^k, \nabla f(x^k)\rangle}_{\text{minimized for } x=x^k-\gamma\nabla f(x^k)} + \frac{1}{2\lambda}\|x - x^k\|_2^2 \tag{4.215}$$

$$= \min_{\gamma\in\mathbb{R}} \langle x^k - \gamma\nabla f(x^k) - x^k, \nabla f(x^k)\rangle + \frac{1}{2\lambda}\|x^k - \gamma\nabla f(x^k) - x^k\|_2^2 \tag{4.216}$$

$$= \min_{\gamma\in\mathbb{R}} -\gamma\|\nabla f(x^k)\|_2^2 + \frac{\gamma^2}{2\lambda}\|\nabla f(x^k)\|_2^2 \tag{4.217}$$

$$= \min_{\gamma\in\mathbb{R}} \|\nabla f(x^k)\|_2^2(\frac{1}{2\lambda}\gamma^2 - \gamma) \tag{4.218}$$

$$= \min_{\gamma\in\mathbb{R}} \frac{1}{2\lambda}\gamma^2 - \gamma \tag{4.219}$$

The last expression is a simple convex quadratic form whose derivative can be equated to zero:

$$\frac{1}{\lambda}\gamma - 1 = 0 \tag{4.220}$$

$$\gamma = \lambda \tag{4.221}$$

This simple demonstration allows us to give a new interpretation of gradient descent algorithm as proximal point methods over linearized versions of the original functional. It also gives us a new point of view to interpret the meaning of the step size of the gradient descent: it mitigates the locality term weight with the linearized estimate fidelity term in the minimization problem. In the framework of a linear least square, this stepwise proximal point methods will yied good results if the objective ressemble its local linear estimate, hence is not too convex.

4.6 Optimality certificate for least square

4.6.1 Introduction

Assuming M is a linear operator whose matrix is of size $n \times k$, we recall that the linear least square reads:

$$\underset{x\in\mathbb{R}^k}{argmin} \quad \frac{1}{2}\|Mx - y\|_2^2 \tag{4.222}$$

It is really interesting to notice that, using the Chambolle-Pocke proximal splitting framework exposed in [**chambolle2011first**] and popularized in the field of tomography by Sidky

et Al in [**sidky2012convex**] the author introduced an algorithm that allows to compute a solution to both the primal and the dual problem. As strong duality holds in such case, monitoring the primal objective, and the dual objective value, allows to compute the primal-dual gap. A PD gap numerically close to zero provides what we can consider as an optimality certificate, which is a very interesting methodology tool in order to compare various imaging model for instance.

In order to apply this method on the simple linear, least square, we will have first to identify the various elements in order to set up our Chambolle Pock framework.

4.6.2 Proximal splitting framework

First, we will use the following formulation for the Chambolle Pock proximal splitting method:

$$\underset{x\in\mathbb{R}^k}{argmin} \quad f(x) + g(Ax) \tag{4.223}$$

Where f and g should be convex functionals. In our case, we identify f as a trivial function: $f(x) = 0$, and g as $g(x) = \frac{1}{2}\|x - y\|_2^2$ In this case, the Fenchel-Rockafellar theorem shows that one can solve the following dual problem:

$$\underset{u\in\mathbb{R}^n}{argmax} \quad -f^*(-L^*u) - g^*(u) \tag{4.224}$$

$$\Leftrightarrow \underset{u\in\mathbb{R}^n}{argmin} \quad f^*(-L^*u) + g^*(u) \tag{4.225}$$

4.6.3 Chambolle Pock algorithm

In order to solve the problem exposed earlier, we will use the Chambolle-Pock strategy which reads:

Take an initial estimates x^0 and u^0 of the primal and dual solutions, a parameter $\tau > 0$, a second parameter $\sigma > 0$ such that $\sigma\tau\|A\|^2 < 1$, and a relaxation parameter $0 < \rho < 2$, and iterates, for $k = 1, 2, \ldots$:

$$u^k = \text{prox}_{\sigma g^*}(u^{k-1} + \sigma L(\tilde{x}^{k-1}) \tag{4.226}$$

$$x^k = \text{prox}_{\tau f}(x^{k-1} - \tau L^* u^k) \tag{4.227}$$

$$\tilde{x}^k = x^k + \rho(x^k - x^{k-1}) \tag{4.228}$$

$$\tag{4.229}$$

Where, x^k converges to a primal solution $x^\star$ and u^k converges to a dual solution $u^\star$.

4.6.4 Deriving the convex Conjugate

4.6.4.1 Convex conjugate of f

We recall that we would like to instanciate the Chambolle-Pock scheme for f a trivial function: $f(x) = 0$. The convex conjugate of f reads:

$$f^*(u) = \max_z \quad \langle u, z\rangle_{\mathbb{R}^n} \tag{4.230}$$

This function has a non finite value $(+\infty)$ for every non zero value of u. Such function reduces to the constraint $u = 0$ that translate into the indicator function of the $\vec{0}$ vector : $\delta_0(u)$

4.6.4.2 Convex conjugate of g

We recall that we would like to instanciate the Chambolle-Pock scheme for g as $g(x) = \|x-y\|_2^2$. The convex conjugate of g reads:

$$g^*(u) = \max_z \quad \langle u, z\rangle_{\mathbb{R}^n} - \frac{1}{2}\|z - y\|_2^2 \tag{4.231}$$

$$\tag{4.232}$$

Where $c(z) = \langle u, z\rangle_{\mathbb{R}^n} - \frac{1}{2}\|z - y\|_2^2$ is a nice concave function that is differentiable, let's see where its derivative vanishes:

$$\frac{\partial c}{\partial z} = 0 \tag{4.233}$$

$$\frac{\partial\langle u, z\rangle}{\partial z} - \frac{1}{2}\left(\frac{\partial\langle z, z\rangle}{\partial z} + \frac{\partial\langle y, y\rangle}{\partial z} - 2\frac{\partial\langle z, y\rangle}{\partial z}\right) = 0 \tag{4.234}$$

$$u - z + y = 0 \tag{4.235}$$

$$z = u + y \tag{4.236}$$

Now that we have found the optimum, we can express the convex conjugate $g^*(u)$:

$$g^*(p) = c(u+y) \tag{4.237}$$
$$= \langle u, u+y\rangle - \frac{1}{2}\|u+y-y\|_2^2 \tag{4.238}$$
$$= \|u\|_2^2 + \langle u, y\rangle - \frac{1}{2}\|u\|^2 \tag{4.239}$$
$$= \frac{1}{2}\|u\|^2 + \langle u, y\rangle_{\mathbb{R}^n} \tag{4.240}$$

4.6.5 Deriving the proximity operator of g^*

The proximity operator of g reads:

$$prox_{\gamma g^*}(u) = \underset{z}{argmin} \quad \frac{1}{2\gamma}\|u-z\|_2^2 + \frac{1}{2}\|z\|^2 + \langle z, y\rangle_{\mathbb{R}^n} \tag{4.241}$$

Where $d(z) = \frac{1}{2\gamma}\|u-z\|_2^2 + \left(\frac{1}{2}\|z\|^2 + \langle z, y\rangle_{\mathbb{R}^n}\right)$ is a nice convex function that is differentiable, let's see where its derivative vanishes:

$$\frac{\partial d}{\partial z} = 0 \tag{4.242}$$
$$\frac{1}{2\gamma}\left(\frac{\partial\langle u,u\rangle}{\partial x} + \frac{\partial\langle z,z\rangle}{\partial x} - 2\frac{\partial\langle u,z\rangle}{\partial x}\right) + \frac{1}{2}\frac{\partial\langle z,z\rangle}{\partial x} + \frac{\partial\langle z,y\rangle}{\partial x} = 0 \tag{4.243}$$
$$\frac{z-u}{\gamma} + z + y = 0 \tag{4.244}$$
$$\left(\frac{\gamma+1}{\gamma}\right)z - \frac{1}{\gamma}u + y = 0 \tag{4.245}$$
$$z = \frac{u-\gamma y}{\gamma+1} \tag{4.246}$$

Now, we have the following proximity operator:

$$prox_{\gamma g^*}(u) = \frac{u-\gamma y}{\gamma+1} \tag{4.247}$$

4.6.6 Wrapping up

We are now able to give the dual problem of the original least square problem:

$$\max_{u\in\mathbb{R}^n} \quad -f^*(-A^*u) - g^*(u) \tag{4.248}$$

$$\max_{u\in\mathbb{R}^n} \quad -\delta_0(-A^*u) - \frac{1}{2}\|u\|^2 - \langle u, y\rangle_{\mathbb{R}^n} \tag{4.249}$$

$$\max_{u\in\mathbb{R}^n} \quad -\frac{1}{2}\|u\|^2 - \langle u, y\rangle_{\mathbb{R}^n} \quad \text{such that} \quad A^*u = 0 \tag{4.250}$$

$$\tag{4.251}$$

A really interesting property for the meticulous scientist, is that we can now actually measure the primal-dual gap for the current set of primal-dual solution:

$$PD(x,u) = \|Ax - y\|_2^2 + \frac{1}{2}\|u\|^2 + \langle u, y\rangle_{\mathbb{R}^n} \tag{4.252}$$

A primal-dual gap numerically close to zero can be considered as an optimality certificate for the current set of primal/dual solution.

This methodological tool is interesting from a theoretical point of view, because it actually allows to properly benchmark different objective functions for their ability to recover a signal, although in many case, one usually compares two methods with the same metric, but with non certified solutions.

Unfortunately, we implemented and tested this method, but even on our low dimensional inverse problems, the primal dual gap never attained a value numerically close to the machine precision, precluding us from generalizing the use of this tool for our studies.

4.7 Conclusion

We saw that there is a large variety of first order methods able to address linear equalities problems, as well as least square problems. Many variations of these algorithms have been recently updated with new expected convergence results in the framework of stochastic optimization.

Unfortunately Assessing the behaviour of all these methods for all combinations of forward/backward projectors models, while varying the reconstructed volume resolution, and the acquisition process (dose, resolution, number of view) would result in a combinatorially complex problem.

In a future work, it would be interesting to setup those various algorithm with all projector combinations, in order to assess their stability, and their convergence speed, as an implicit surrogate for assessing the relative underlying linear system condition number.

It would also be interesting to derive a primal-dual forward-backward scheme in order to setup an optimality certification method for he least square that would probably converge faster in practice.

Chapter 5: A Sparse Model for Tomographic Reconstruction

Sommaire

5.1 Introduction

Although previous chapters where dedicated to study the basics of tomography, such as tomographic system model, and optimization for model fitting, we did not used really advanced

apriori on the solution. However, in the framework of ill-posed problem, or linear problems with a high dimensinal nullspace, one needs to exploit prior information in order to regularize the inverse problem, and find an interesting solution.

One of the most generic prior that have been designed was the Tickonov minimum norm solution regularizer, but in the recent years, with the advent of versatile proximal splitting framework for non-smooth optimization, and high performance parallel computers, more and more sophisticated priors have been successfully exploited.

In this chapter, we will mostly restrict our attention to the class of sparse priors. Sparse priors have been extensively used, for the past few decades as a regularizing tool for common inverse problems in imaging. The literature about design of efficient sparsifying transform and sparsity promoting algorithm is huge, and it would be difficult to establish an exhaustive list of all approaches that have been applied to the problem of CT reconstruction. Instead we will focus on one specific tool, arising from the field of harmonic analysis, known as dual-tree complex wavelet transform (DT$\mathbb{C}$WT).

Our aim here, is to try to overcome some limitations of well-known sparsifying transform, like the loss of texture informations often attributed to the total variation model or the lack of directionality of some wavelet with high order vanishing moments like Daubechies'.

A previous study with a similar approach has been conducted in [**vandeghinste2013iterative**], and showed that shearlet transform yielded better results that TV, but only in a 2D setting. Although extensions of ridgelet, curvelets, shearlets, ... to 3D have been studied, for instance in [**kutyniok2012optimally**], and provides optimal sparsity properties for some class of functions, to our knowledge, frame based sparsity prior in CT reconstruction were mostly restricted to 2D structural informations.

It should be noticed that, in the framework of 3D imaging, lack of directionality of common real wavelets and computational cost of non separable wavelets become increasingly challenging obstacles to practical use.

As a consequence, the extension of the frame based sparsity prior to 3D, although computationally demanding, seems to be an interesting lead, in this work, we will address the DT$\mathbb{C}$WT as a numerically efficient 3D separable transform.

More precisely, we will challenge the the DT$\mathbb{C}$WT transform in the framework of a simple CBCT reconstruction algorithm, in order to see if the directionality and shift invariance features can be efficiently leveraged despite of the inherent redundancy.

5.2 Sparsity in signal processing

5.2.1 Terminology

5.2.1.1 Frames and dictionaries

Based on [**vetterli1995wavelets**] and [**starck2010sparse**] we provide the following definitions, that will be used throughout this chapter :

Frames or Riesz Basis
A frame of a vector space V with an inner product can be seen as a generalization of a basis to sets which may be linearly dependent. A very good introduction on this topic can be found in chapter 5 of [**vetterli1995wavelets**].

The definition states that a sequence p_k in a Hilbert space H is a frame if there exist numbers $\mu, \sigma > 0$ such that

$$\forall x \in H, \mu\|x\|^2 \leq \sum_k |\langle x, p_k \rangle|^2 \leq \sigma\|x\|^2 \tag{5.1}$$

The numbers μ, σ are called frame bounds. The frame is tight is $\mu = \sigma$, and in case all the vectors p_k are of unit length, μ gives the redundancy factor of the frame, and the following expansion holds:

$$y = \mu^{-1} \sum_k \langle y, p_k \rangle x_k \tag{5.2}$$

As the family of p_k may not be linearly independant, this expansion may not be unique, but is the one that has the minimal norm, the $\mu^{-1}p_k$ family is also called the minimal dual synthesis frame, this minimal norm property can be easily shown using Moore-Penrose pseudo inverse property, see chap 8.2 of [**starck2010sparse**].

If the frame is tight and $\mu = 2$ in the case of unit vectors, we can say that the sequence of vectors p_k contains 2 times more vectors that necessary to span V.

It can be noticed, that a union of orthogonal basis of V automatically forms a tight frame, but in the more general case, proving that a frame is tight, or proving that it is numerically close to a tight frame, requires to show that all its nonzero singular values are equal, or respectively close to each other, see [**eldar2002optimal**].

Analysis and Synthesis for a frame

Assuming G and H are Hilbert spaces, we define the analysis operator as $T : G \to H$ given by $T(x) = \langle x, p_k \rangle_k$.
The fact that $T(x) \in H$ is a bounded linear operator follows from the frame inequality.
The synthesis operator, being the adjoint of T, will be denoted by $T^\star : H \to G$ and is given by $\alpha_k \to \sum_k \alpha_k p_k$.
A corollary of the previous statements is that, in the case of a tight frame, we have $TT^\star = \mu Id$

Atom
An atom is a general concept that refer to an elementary waveform which can be used as building blocks, to construct more complex signals by linear superposition.

Dictionary
A dictionary Φ is an indexed collection of atoms $(\phi_k)_{k=1,\ldots,M}$ In the framework of discrete-time, finite-length signal processing, a dictionary could be viewed as an $N * M$ matrix whose columns are discrete atoms of size N.
When the dictionary has more columns than rows, $M > N$, it is called *overcomplete* or *redundant*, but there is no requirement related to the operator norm of Φ or the distribution of its singular values like in the frame basis.

Analysis and Synthesis for a Dictionary
Analysis is the operation that associates with each signal x a vector of coefficients α such that: $\alpha = \Phi^\intercal x$.
Synthesis is the operation of reconstructing as signal x by superimposing atoms through the matrix vector operation: $\Phi\alpha$.

In the overcomplete case, inverting the synthesis operator for a known signal x amounts to the resolution of $x = \Phi\alpha$ which could potentially lead to an underdetermined system of linear equations, i.e, finding α could possibly yield an infinite number of solution.

5.2.1.2 Sparsity

Strictly Sparse Signals
A signal x, considered as a vector in a finite dimensional subspace of $\mathbb{R}^N, x = [x[1], \ldots, x[N]]$ is strictly or exactly sparse if most of its entries are equal to zero, ie, if its support $\Lambda(x) = 1 \leq i \leq N | x[i] \neq 0$ is of cardinality $k \ll N$

A k-sparse signal is a signal for which exactly k samples have nonzero value. If a signal is not sparse, it may be *sparsified* in an appropiate transform domain.

Compressible Signals
Signals of practical interest are rarely strictly sparse, but they may be *compressible* or *weakly sparse* if the sorted magnitude $|\alpha_{(i)}|$ of the representation coefficients $x = \sum_i \alpha_i \phi_i$ decays rapidly according to the power law

$$|\alpha_{(i)}| \leq R.i^{-\frac{1}{p}}, i = 1, \ldots, n-1 \tag{5.3}$$

The previous expression helps to define the weak l^p norm: the smallest R such that this inequality holds, which is also the radius of the weak l^p ball that contains the vector x.

In [**candes2006compressive**] it is also recalled that compressible signals can also be caracterized by the approximation error defined as the $l2$ norm of the difference between a signal x and its best k-term approximation using k coefficients (denoted x_k) in the basis Φ which decays as :

$$\|x - x_k\|_2 \leq C.R.k^{\frac{1}{2}-\frac{1}{p}} \tag{5.4}$$

In a previous work [**notargiacomo:16:sro**], using this best k-term approximation method empirically, we showed that DT$\mathbb{C}$WT with a sufficiently high number of scales provided a good approximation of the weak l^p ball model for a CT medical image.

5.2.1.3 Synthesis and Analysis formulation of sparse coding

In [**selesnick2009signal**], the authors performed an interesting comparison between two formulations of the l_1 relaxation of the sparse regression problem: the synthesis version, with S a synthesis operator:

$$\min_x \|y - HSw\|_2^2 + \lambda \|w\|_1 \tag{5.5}$$

and the analysis version, with A an analysis operator:

$$\min_x \|y - Hx\|_2^2 + \lambda \|Ax\|_1 \tag{5.6}$$

In their work, the authors used classical proximal splitting framework to study both formulations. They suggested that analysis prior may be the most appropriate one to tackle the problem of signal retrieval, based on arguments related to the lack of sparsity of the low frequency elements in wavelet basis, and interpreting the success of total variation.

More recently, in [**pustelnik2012relaxing**] the authors developed an elegant proximal splitting framework able to handle both analysis and synthesis formulation while relaxing the tight frame condition. This approach addresses the problem of prefiltered wavelet tree that are no more fully orthogonal, like the one we will use in this paper. Although a bit less general than the Chambolle-Pock method [**chambolle2011first**], the authors claim that their method, dedicated to 'quasi' tight frame has a faster convergence.

5.2.2 Sparsity and compressive sensing in CT

Although exactly sparse signals are of limited interest in real cases, good approximation of k-sparse signals can be derived from weakly sparse signals.

The concept of best k-term approximation in particular has been used in the framework of compressed sensing in [**cohen2009compressed**], where theoretical bounds have been proposed regarding the performance of compressed sensing systems.
The author addresses the following problem:
For a given norm $\|.\|_X$ and $k < N$, what is the minimal number of measurements n to be made, for which exists a pair of encoder/decoder (Φ, Δ) such that

$$\|x - \Delta(\Phi x)\|_X \leq C_0 \sigma_k(x)_X \tag{5.7}$$

for all $x \in \mathbb{R}^N$, with C_0 a constant independent of k and N, and $\sigma_k(x)_X$ being the best k-term approximation of x in the norm $\|.\|_X$.
Unfortunately, those bounds rely on restricted isometry property constant of the sampling matrix Φ, which are very difficult to retrieve for a given physical acquisition process model.

In [**joergensen2011toward**], the authors have made an interesting work about the compressive sensing approach in the field of X-Ray Computed Tomography. They tried to infer the best number of views for a given X-Ray flux using the CS framework under the total variation sparsity model.

They suggested that the number of samples predicted by CS theory, originally designed for specific sampling matrices, was far too low for a perfect recovery, and that the same dose of X-Ray used to generates more line integrals measurements resulted in more accurate reconstructions.

More recently in [**jorgensen2014testable**], noiseless recovery condition in fan beam CT case have been studied experimentally on a set of simulated images with a rigorous methodology. Both uniqueness and recoverability have been tested for direct space $l1$ norm, anisotropic and isotropic total variation sparsity model, and a sharp phase transition diagram has been obtained in every case.

This study, as well as a previous one, including poisson-like noise robustness [**jorgensen2012empirical**], showed that some of the results of CS, namely phase transition and uniqueness of recovery where applicable to fan beam CT for sparse signals in the direct space. These results are quite promising considering that, to our knowledge, neither restricted isometry, nor incoherence or isotropy properties [**candes2011probabilistic**] have ever been proved to be met by the sampling/encoding scheme imposed by any CT acquisition system.

Still, the role of weakly sparse signal for CT acquisitions in the CS framework remains unclear, and current results suggest that in real world acquisition systems, compressive sensing may not be for now, the best way to improve the tradeoff between X-Ray dose and image quality.

Hence the success of numerous statistical approaches that seems to handle noise inconsistency in addition to sparsity as regularizing prior, for instance [**ramani2012splitting**] and [**mcgaffin2015fast**].

Although it seems difficult to give a clear interpretation to the success of sparse prior in CT reconstruction using the Compressive Sensing framework, it seems that some work still remains for assessing the relevance of various sparsifying transform and morphological diversity concepts to the field of CT image reconstruction, see [**pan2009commercial**]

5.2.3 Dual-Tree Complex Wavelet transform

Among all existing sparsifying transform, that have already been successfully used in various field of imaging, we propose here to study the dual-tree complex wavelet transform. The idea of this work is, as exposed in [**vandeghinste2013iterative**], to try to overcome some limitations of the total variation model, like the loss of texture informations, or the lack of directionality of some wavelet with higher order vanishing moments like Daubechies'.

Although extensions of curvelets and shearlets [**kutyniok2012optimally**] to 3D have been studied, and provide very interesting optimality properties for the class of smooth objects with discontinuities along C^2 curves, we will restrict our study to the DT$\mathbb{C}$WT here.

5.2.3.1 Interesting properties of the DT$\mathbb{C}$WT

The rational behind the design of DT$\mathbb{C}$WT transform is to overcome some drawbacks of real and separable wavelets. Here is a short summary of the main points exposed in [**selesnick2005dual**]:

Oscillations
Singularities of signals in direct space may not lead to large coefficients in the wavelet domain but instead to oscillating patterns which may have to interfere in order to recreate the initial discontinuity.

Shift Variance
Small shift in direct space may have dramatic consequences over the oscillating patterns generated in the wavelet domain.

Aliasing
The down-sampling of signals and subsequent filtering during the analysis process may generate aliasing during synthesis if coefficients have been slightly modified, this is also why undecimated wavelets and 'a trou' algorithm have been designed, although they also comes with an important redundancy factor.

Lack of directionality
The separability of real filters used in n-dimensional wavelets generates checkerboard patterns that preclude from efficiently discriminating ridge in 2D-3D imaging. This is also why anisotropic and non-separable wavelets has been designed ([**candes1999ridgelets**], [**starck2002curvelet**]).

We may add that in the framework of 3D imaging, lack of directionality of common real wavelets and computational cost of non separable wavelets became an increasingly annoying obstacle to their use. The DT$\mathbb{C}$WT do offers a good tradeoff between its interesting properties, and computational cost which make it a transform of great practical interest.

Practical use of 3D DT$\mathbb{C}$WT for video denoising has been studied in [**selesnick2003video**], and outperformed classical separable wavelet although it appeared that 2D directionality was more relevant for modeling 2D signal than its extension to 3D was for 2D+time signals. Successfull use of 2D DT$\mathbb{C}$WT has also been reported in fingerprint reconstruction tasks [**rameshkumar20122d**], whose visual appearance resemble microstructures observed inside human bones along various orientations.

5.3 Proposed approach for sparse regression

5.3.1 Instanciating our sparse regression algorithm

5.3.1.1 The analysis formulation

Authors in [**selesnick2009signal**] propose to use the Chambolle-Pock algorithm for solving the analysis formulation, which is a very good choice in a general dictionary framework, but we decided in a first approach to develop our method using the simple forward-backward splitting scheme.

The forward-backward splitting technic and its accelerated version for sparse regression: Fista, see [**beck2009fast**], is generally used to solve the synthesis formulation, but it is easy to set up in both formulations when the terms $\|w\|_1$ and $\|Ax\|_1$ have a simple proximity operator. Assuming that our complex wavelet basis is a tight frame, we will see that the proximity operator of $\|Ax\|_1$ can be easily computed.

5.3.1.2 Proximity operators involving tight frames and complex valued coefficients

As seen in 5.3.1.1, we are interested in computing the proximity operator $prox_{f\circ T}$ of an operator f composed with the analysis operator T of a tight frame. In our analysis formulation, we have $f(y) = \|y\|_1$ which is the l_1 norm, and, for convenience, we write $T(x) = Tx$, where T is a redundant wavelet tranform, that forms a tight frame.

Proximal calculus rules gives us the following equivalence: Given H and G two Hilbert spaces, f is a convex, lower semi continuous, and proper function from G to $\mathbb{R}$ and T a bounded linear operator from H to G such that $T^\star T = TT^\star = \mu Id$ with $\mu \in]0, +\infty[$, then

$$prox_{f\circ T} = Id - \mu^{-1} T^\star \circ (Id - prox_{\mu f}) \circ T \tag{5.8}$$

In order to use that property in the framework of complex sparsity that arise with complex wavelets, we must recall the expression of the l_1 norm for complex-valued vector:

$$\|x\|_1 = \sum_{i=0}^{N-1} \sqrt{Re(x_i)^2 + Im(x_i)^2} \tag{5.9}$$

And the related proximity operator, which can simply be derived from the real case, see [**maleki2013asymptotic**] :

$$prox_{\tau f} = \eta(x, \tau) \tag{5.10}$$

$$\text{with } \eta(a+ib, \tau) = \left(a + ib - \frac{\tau(a+ib)}{\sqrt{a^2+b^2}}\right) \mathbb{1}_{a^2+b^2>\tau^2} \tag{5.11}$$

5.3.1.3 Forward Backward splitting instance

The complex wavelet framework developed in [**selesnick2005dual**] let some room for interpreting the data in the wavelet domain, either as complex valued coefficients multiplying real wavelets, or pair of real coefficients multiplying real and imaginary wavelets. In our proximal splitting framework, we chose the proximity operator over complex valued coefficients but an equivalent derivation can be made in the framework of a 2-group-sparsity for real valued coefficients.

Let recall the expression of the sparse analysis problem:

$$\min_x g(x) + h(x) \tag{5.12}$$

$$\min_x \|y - Hx\|_2^2 + \lambda \|Tx\|_1 \tag{5.13}$$

The high level forward backward splitting iterations reads:

$$x^{(k+1)} = prox_h \left(x^{(k)} - \gamma \nabla g(x^{(k)})\right) \tag{5.14}$$

and we can derive the expression of the proximity operator of $h(x) = f \circ T(x)$, in case it is a composition of two operators:

$$prox_h = prox_{\lambda f \circ T} \tag{5.15}$$

$$= Id - \mu^{-1} T^\star \circ (Id - prox_{\mu\lambda f}) \circ T \tag{5.16}$$

$$= Id - \mu^{-1}(T^\star T - T^\star \circ prox_{\mu\lambda f} \circ T) \tag{5.17}$$

As we know that $T^\star T = \mu Id$, expression simplifies

$$prox_{\lambda f \circ T} = \mu^{-1} T^\star \circ prox_{\mu\lambda f} \circ T \tag{5.18}$$

Let's now imagine that we define T', a scaled version our operator T with scaling factor $\mu^{-\frac{1}{2}}$ such that

$$T'T'^{\star} = \mu^{-\frac{1}{2}} T \mu^{-\frac{1}{2}} T^{\star} \tag{5.19}$$
$$= \mu^{-1} T T^{\star} \tag{5.20}$$
$$= \mu^{-1} \mu Id \tag{5.21}$$
$$= Id \tag{5.22}$$

The expression simplifies even more because the new μ' frame bound is equal to 1, and we get :

$$prox_{\lambda f \circ T'} = T'^{\star} \circ prox_{\lambda f} \circ T' \tag{5.23}$$

Our forward-backward splitting instance then reads:

$$x^{(k+1)} = T'^{\star} prox_{\lambda h} \left(T' \left(x^{(k)} - \gamma \nabla g(x^{(k)}) \right) \right) \tag{5.24}$$

Where λ is the regularization weighting parameter, γ is chosen as the spectral radius of $H^t H$ in equation 5.12, and $prox_{\lambda h}$ is simply the complex soft thresholding operator defined in equation 5.3.1.2.

5.3.2 Choice of the 3D-DT$\mathbb{C}$WT

Instantiating and implementing a separable complex wavelet transform requires first a basic understanding of the relation between multiresolution analysis and filterbanks, that has been first exposed by Mallat in [**mallat1989theory**].

The specific case of filterbank design for complex wavelets has been tackled in [**selesnick2005dual**], especially the method to obtain nearly analytic wavelets (without negative frequencies), which is the key feature for high directionality.

According to this work, we used the pair of Q-shift filters based solution to generate the half sample delay as a Hilbert transform. This methods mimics the 90° phase shift applied in single sideband modulation, and has the nice property of perfect reconstruction, short support (6tap) and orthogonality.

In the framework of convex optimization, fully orthogonal wavelet trees at every stage are desirable, in order to form a tight frame of the image space (usually $\mathbb{R}^N$), so that we can use the tools described in 5.3.1.3.

In practice we decided to give up on the first scale orthogonality in order to favor wavelet analycity, by using the (9,7) bi-orthogonal Antonini filter at the first stage as described in [**selesnick2005dual**], without significant convergence problem.

	ψ_0	ψ_1	ψ_2	ψ_3	ψ_4	ψ_5	ψ_6	ψ_7
Z-Axis	ψ_h				$j\psi_g$			
Y-Axis	ψ_h		$j\psi_g$		ψ_h		$j\psi_g$	
X-Axis	ψ_h	$j\psi_g$	ψ_h	$j\psi_g$	ψ_h	$j\psi_g$	ψ_h	$j\psi_g$
Real Part	+1	0	0	-1	0	-1	-1	0
Imaginary Part	0	+1	+1	0	1	0	0	-1

Table 5.1: Table of 3D complex wavelet octree components

It should be noticed that directionality given by the nearly analytic behaviour of $\mathbb{C}$WT comes with a redundancy ratio of greater importance with the growth of the number of spatial dimension D as 2^D, which gives the 3D DT$\mathbb{C}$WT a redundancy factor of 8.

5.3.3 Implementation of the 3D-DT$\mathbb{C}$WT

In order to take advantage of recent advances in high performance computing, we implemented our own GPU version of the n-D DT$\mathbb{C}$WT using Cuda. But in order to understand the various computational and memory requirements this wavelet transform relies on, one must get back to its definition.

In the framework of complex wavelet transform that has been developed in [**selesnick2005dual**], the author basically defines a complex function in the direct space as follows:

$$f(x) = \psi_h(x) + j\psi_g(x) \tag{5.25}$$

This expression stands for either a wavelet or a scaling element, although in both cases, the real function ψ_g should be the best possible approximation of the Hilbert transform of $\psi_h(x)$, for the purpose of negative frequencies cancellation.

This 1-dimensional scheme can be extended to the 3D case, where we want to define a separable function from three 1D functions as defined in eq 5.25. The separability property allows us to derive a scalar value in every point of a 3D domain from the tensor product of three 1D function. For the sake of our argument here, we will consider that those three 1D functions are virtually extended to 3D although their value only varies along one axis, here denoted by x,y or z, in order to be able to use the simple product:

$$f(x,y,z) = [\psi_h(x) + j\psi_g(x)][\psi_h(y) + j\psi_g(y)][\psi_h(z) + j\psi_g(z)] \tag{5.26}$$

We can develop and rearrange this product into the sum of 8 terms $\psi_0, \psi_1, \ldots, \psi_7$, that we present on Table 5.1.

The successive dual-filtering steps along each direction that led to this construction was explicited on the figure 5.1, where h_0 and h_1 stands respectively for the low pass projection (scaling) and high pass projection (wavelets).

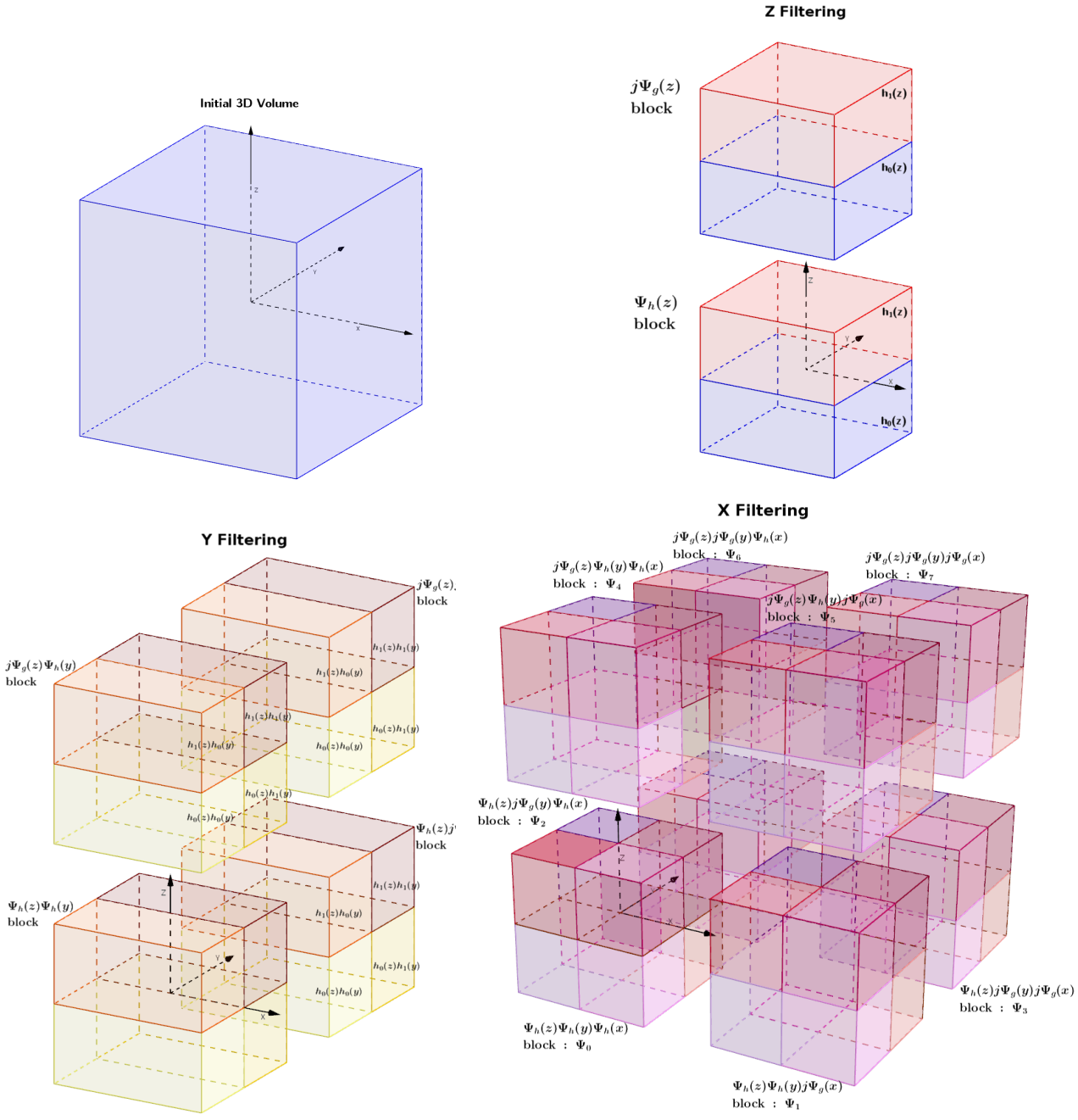


Figure 5.1: The multiple filtering steps of the dual tree complex wavelet transform analysis operator

Now, we can notice that the eight 3D elements in table 5.1 can now be grouped in two

	Real part	Imaginary part
$\psi(x)\psi_(y)\psi(z)$	$\psi_0 - \psi_3 - \psi_5 - \psi_6$	$\psi_1 + \psi_2 + \psi_4 - \psi_7$
$\psi(x)\psi(y)\overline{\psi(z)}$	$\psi_0 - \psi_3 + \psi_5 + \psi_6$	$\psi_1 + \psi_2 - \psi_4 + \psi_7$
$\psi(x)\overline{\psi(y)}\psi(z)$	$\psi_0 + \psi_3 - \psi_5 + \psi_6$	$\psi_1 - \psi_2 + \psi_4 + \psi_7$
$\psi(x)\overline{\psi(y)\psi(z)}$	$\psi_0 + \psi_3 + \psi_5 - \psi_6$	$\psi_1 - \psi_2 - \psi_4 - \psi_7$

Table 5.2: Table of separable mixture of complex wavelet which covers all 4 octants of the positive x-axis frequencies orthant

categories:

- The real elements:

$$\psi_0 - \psi_3 - \psi_5 - \psi_6 \tag{5.27}$$

- The imaginary elements:

$$\psi_1 + \psi_2 + \psi_4 - \psi_7 \tag{5.28}$$

Adding all real elements of the octree together, and all imaginary elements together to get only one complex function won't enable us to exploit the interesting directionality property we are looking for. Indeed, canceling the negative part of the spectrum in every direction x,y,z will only allow us to select the fully positive octant in the 3D spectrum.

Instead, we recall that we are interested in alternating the negative frequency cancellation property using:

$$\psi_+(x) = \psi_h(x) + j\psi_g(x) \tag{5.29}$$

and the positive frequency cancellation property using:

$$\psi_-(x) = \overline{\psi_+(x)} = \psi_h(x) - j\psi_g(x) \tag{5.30}$$

Using this scheme alternatively over various directions in a 3D setting allows us to define 4 complex and somehow analytic wavelets, that will be covering 4 octants of the frequency space. In our case, those four octant are arbitrarily chosen to be in the positive x-axis frequencies orthant. That way, by central symmetry, all 8 octants of the frequency would be reachable by setting the real or imaginary coefficients accordingly.

The combination that allows for a proper 4-octant partitioning of the positive x-axis frequencies orthant are presented in 5.2, from this Table, we can derive the transformation, that will help us to generate the real and imaginary parts of the 4 octants described earlier, from the "'raw filtered octants"' $\psi_i, i \in 0, 1, \ldots 7$:

$$\begin{pmatrix} Re(\psi(x)\psi_(y)\psi(z)) \\ Im(\psi(x)\psi_(y)\psi(z)) \\ Re(\psi(x)\psi(y)\overline{\psi(z)}) \\ Im(\psi(x)\psi(y)\overline{\psi(z)}) \\ Re(\psi(x)\overline{\psi(y)}\psi(z)) \\ Im(\psi(x)\overline{\psi(y)}\psi(z)) \\ Re(\psi(x)\overline{\psi(y)\psi(z)}) \\ Im(\psi(x)\overline{\psi(y)\psi(z)}) \end{pmatrix} = M_{OctToCpx} \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \\ \psi_3 \\ \psi_4 \\ \psi_5 \\ \psi_6 \\ \psi_7 \end{pmatrix} \tag{5.31}$$

Where $M_{OctToCpx}$ reads:

$$M_{OctToCpx} = \frac{1}{4\sqrt{2}} M_{nnOctToCpx} \tag{5.32}$$

$$= \frac{1}{4\sqrt{2}} \begin{pmatrix} 1 & 0 & 0 & -1 & 0 & -1 & -1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & -1 \\ 1 & 0 & 0 & -1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & -1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & -1 & 1 & 0 \\ 0 & 1 & -1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & -1 & 0 \\ 0 & 1 & -1 & 0 & -1 & 0 & 0 & -1 \end{pmatrix} \tag{5.33}$$

The normalization factor $\frac{1}{4\sqrt{2}}$ is actually equal to the product of $\mu^{-\frac{1}{2}} \times s$.

We recall that $\mu^{-\frac{1}{2}} = \frac{1}{\sqrt{8}} = \frac{1}{2\sqrt{2}}$ is the DT$\mathbb{C}$WT 8-redudant tight frame normalization factor presented in 5.3.1.3.
And $s = \frac{1}{\sqrt{4}} = \frac{1}{2}$ is the ratio used to make the matrix $sM_{nnOctToCpx}$, a unitary matrix.

The inverse mapping is then simply:

$$M_{CpxToOct} = M^{-1}_{OctToCpx} = M^{\intercal}_{OctToCpx} \tag{5.34}$$

5.3.4 MultiGPU implementation of the DT$\mathbb{C}$WT

In order to accelerate the successive filtering steps, we worked on the software definition of filtering workloads, that can be distributed in parallel over multiple GPUs. Using a host-centralized memory model was mandatory in our case, indeed simple calculations shows that the expression of the DT$\mathbb{C}$WT coefficients of a 1024^3 image in single precision floating point requires 32GB of memory, precluding for in-memory GPU implementation. Our filtering workloads were basically structures defining a source host address to be copied, a memory layout, related to the filtering direction, and a number of slices to be processed. The workload distribution among GPU relied on openMP library, so that our only task was to tune the size,

in number of slices, of the atomic memory chunk to be processed at the first analysis step, handling full resolution images. Subsequent filtering steps had their chunk size automatically computed in their filtering workloads, so that memory requirements for every step was not higher than memory requirements for the first step.

5.3.5 Textured phantom design

In order to assess the ability of various reconstruction methods to retrieve textured images, we needed a 3D numerical model. In order not to be dependent on the image resolution, and the dataset arbitrariness, we selected our reference volume to be a discrete sampling of the analytically defined Marschner-Lobb function, first presented in [**marschner1994evaluation**] which reads:

$$\rho(x,y,z) = \frac{1 - sin(0.5\pi z) + \alpha(1 + \rho_r(\sqrt{x^2+y^2}))}{2(1+\alpha)} \tag{5.35}$$

Where

$$\rho_r(r) = cos(2\pi f_M cos(0.5\pi r)) \tag{5.36}$$

This function has been extensively used by the computer graphics and visualization community as a benchmark for various 3D interpolation or rendering algorithms. One should notice that this function features adjustable frequency directional pattern, which differs from the piecewise constant regions features in the Shepp-Logan phantom.
We postulate that this model exhibit some of the pattern that can be observed in medium resolution pathologic bone micro-structures images, for instance bone density gradients, in the whole experiment, we used the following parameters $\alpha = 0.25$ and $f_M = 12$.

In order not to favor axis aligned structures in the image, we applied a linear transformation to our coordinate system so that the initial coordinate system undergo a counterclockwise rotation about the vector $(0.5, 0.5, 1)$ by $\frac{pi}{3}$ radians.

A 3D rendering of a binarized version (arbitrary threshold) of the volume we generated using this method is presented on figure 5.2.

5.4 Results

5.4.1 Numerical experiments on synthetic data

5.4.1.1 The noisy recovery usecase

To perform our experiments comparing multiple reconstruction algorithms, we used the ML phantom, sampled over a 3D grid of 256^3 voxels which was then masked using a binary

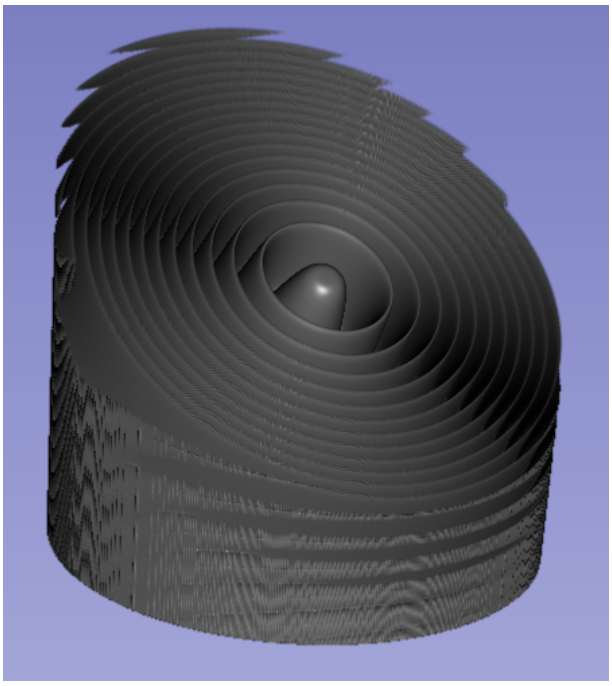


Figure 5.2: 3D rendering of a thresholded version of our Marschner-Lobb numerical phantom

cylinder. We used the reconstruction framework provided by the openRTK software package : [**rit2014reconstruction**] and its various tools to generate and reconstruct the data.

Using the GPU implementation of ray casting projector provided by RTK, and the cone beam geometry module, we generated 200 projections of resolution 256^2, regularly distributed across a circular source-detector trajectory covering 200°.

We then added to those attenuation images, a centered gaussian noise with standard deviation equal to 1% of the mean of the attenuation image stack. We extracted one of the projection image for visualization in 5.3.

5.4.1.2 Comparing sparsity models for CBCT regularization

Our aim in this experiment was to compare our algorithm performance in the context of challenging data inconsistency related to noisy measurements, with two other iterative algorithm exploiting another sparsity a-priori. We chose two different instances of the $L1$ regularized ADMM algorithm provided by openRTK, the first using the total variation linear operator, and the second, orthogonal Daubechies wavelets transform with 5 levels. 5 decomposition levels where also used for the DT$\mathbb{C}$WT.

In order to get the less possible biased overview of the robustness of each algorithm, we performed a logarithmic (base 10) regularization parameter sweep over the best decade for each algorithm. It should be noticed that, the consistency parameter β for the ADMM instances

was arbitrarily set to 200, but we explicitly used 100 iterations for each algorithm in order to minimize the effect of convergence speed not directly related to the main hyper-parameter.

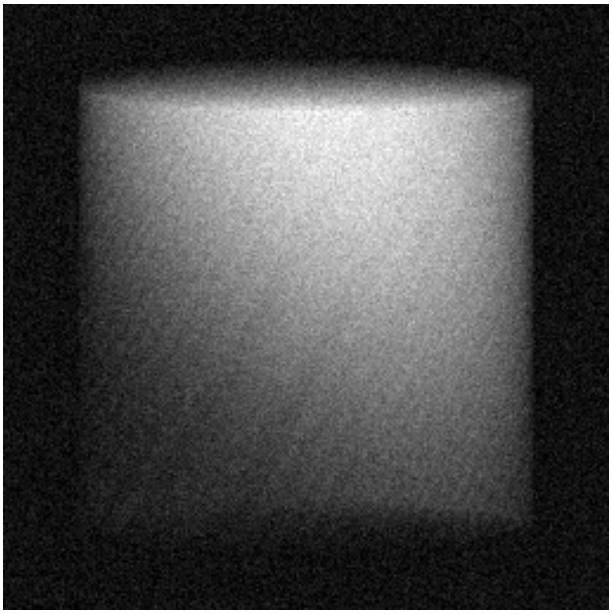


Figure 5.3: Example of a noisy cone beam projection from our Marschner-Lobb numerical dataset

For each reconstruction, we measured the peak signal to noise ratio (PSNR), and the mean 3D structural similarity index (SSIM) computed over 5^3 voxels patches, results are presented on figure 5.4.

We extracted the same axial slice for all reconstruction that got the maximum ssim value along the regularization path, and reported the images on figure 5.5.

In order to analyze qualitatively the behaviour of the 3 algorithms in the case where sparsity was overestimated, we also extracted the same axial slice for reconstruction that where beyond the optimal regularization coefficient value, and reported the images on figure 5.6.

5.4.2 Numerical experiments on real data

5.4.2.1 CBCT reconstruction of a human knee specimen

In an attempt to leverage DT$\mathbb{C}$WT interesting properties for regularizing real CBCT dataset, we designed a simple experiment that was conducted on a homemade rotating platform using a Thales 2630S flat panel detector and a IAE RTC 600 HS 0.6/1.2 X-Ray source. 200 images, binned in 780×720 ($368\mu m$ equivalent pixels) were acquired at 70 kVp and 8mA over a 200° angular range (circular trajectory), no denoising filter was applied nor beam hardening correction and attenuation maps were obtained using a constant I_0 estimation.

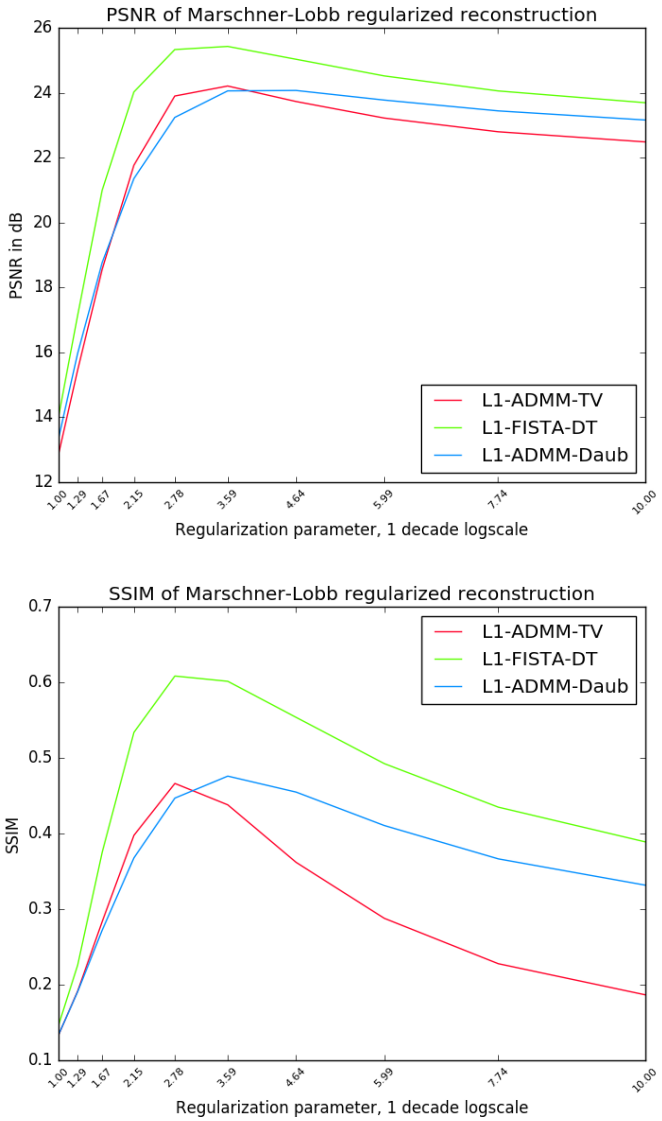


Figure 5.4: Reconstruction quality metrics along regularization parameter

We figured a picture of our setup on figure 5.7

Reconstruction of 512^3 voxels of size $244 \times 244 \times 488 \mu m$, was performed using a single NVidia GTX Titan X, with 100 Fista iterations and a manually chosen regularization parameter. The reconstruction resolution here, was one of the key parameter, as bone microstructure size is usually comprised between 70 and a few hundred of μm, see [**parkinson2013characterisation**], reconstruction with a resolution lower than a few tens of μm tends to suffer from partial volume effect.

For this reason, bone microstructure recovery was a challenging task, although we are looking forward to perform all reconstructions in a higher resolution, we found that 512^3 was

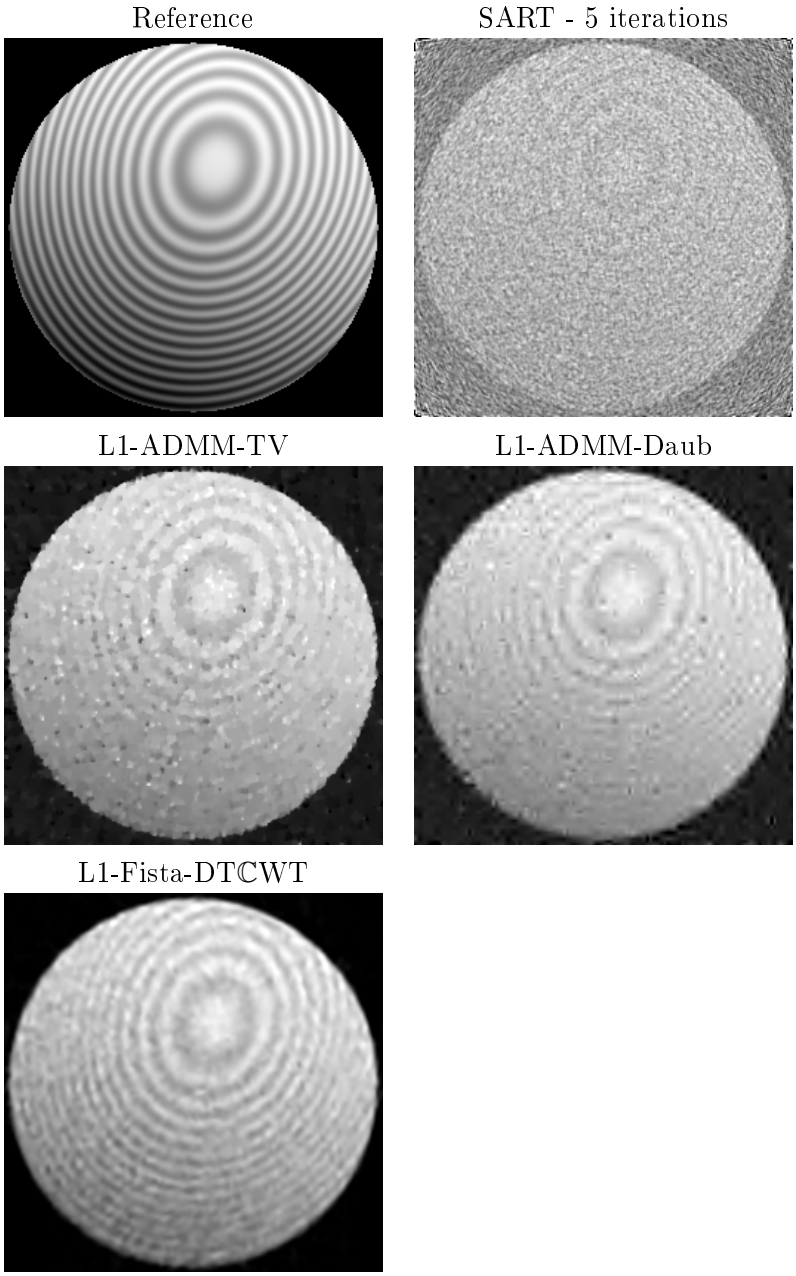


Figure 5.5: Visual overview of an axial slice with 4 different reconstruction methods

a reasonnable tradeoff to start a comparison with multiple methods.

Other methods used here are the same used in 5.4.1.2, 100 iterations were performed each time, but we did not performed an exhaustive regularization parameter sweep due to the size of the dataset and the inherent runtime. We restricted ourselves to manually choosing one set of parameters per algorithm, that used to give visually interesting results in other experiments with the same setup.

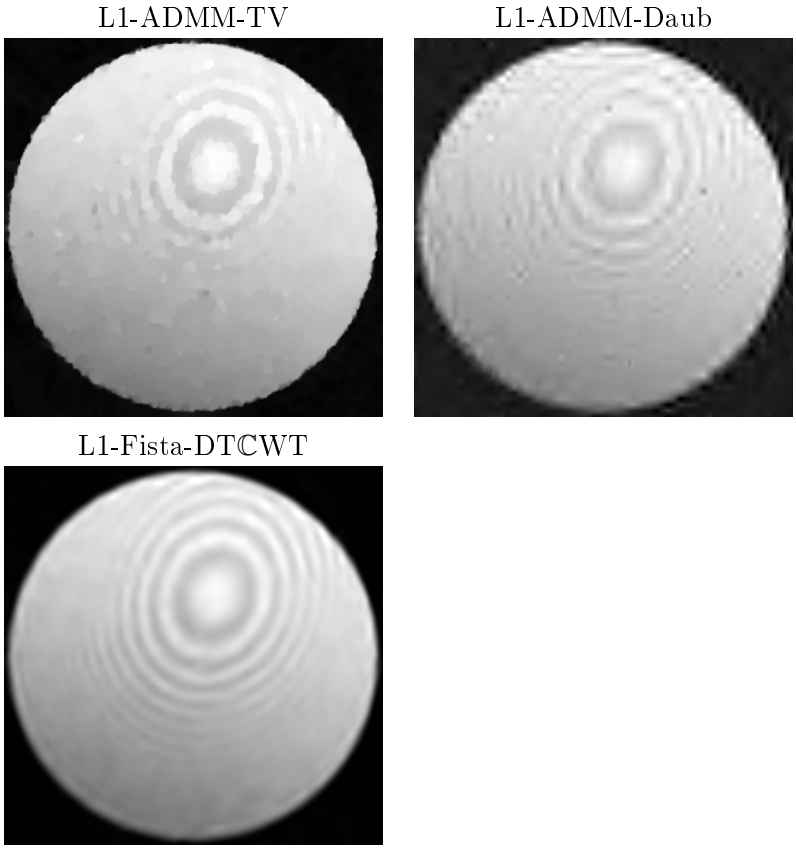


Figure 5.6: Visual overview of the effect of sparsity overestimation for 3 different regularized reconstruction methods

The total run time in our setting was about 18 minutes for DTℂWT based Fista Algorithm, which was approximately 2 to 3 times faster than the ADMM algorithms on the same computer using the same number of iterations.

The data presented in figure 5.8 shows a small ROI of $192 \times 192 \times 1$ voxels taken in an axial slice of the volume that exhibits microstructures, the exact same floating point to 8-bit grayscale image conversion was applied in every case.

The yellow line present on every image from figure 5.8 marks the line along which we extracted the linear profile presented on figure 5.9, in order to highlight the contrast/resolution enhancement enabled by our algorithm.

5.4.2.2 Undersampling strategy

Using the setup presented in the beginning of section 5.4.2.1, we also made acquisition with real frozen human knee, kindly provided by Christine Chappard from Bioingenierie et Bioimagerie Osteo-Articulaire (B2OA) laboratory. This time 360 projections where acquired over

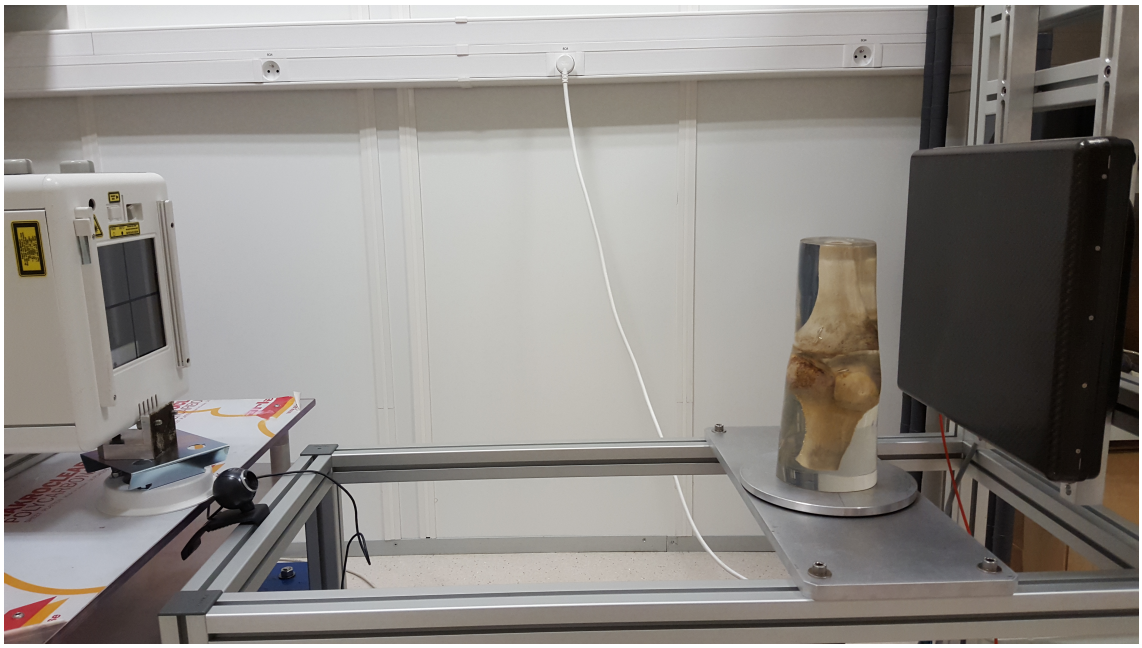


Figure 5.7: Rotating platform equipped with a Thales 2630S FPD, and a IAE RTC 600 HS 0.6/1.2 X-Ray source, with a plastinated knee specimen

360° in order to get a reference reconstruction with a quality dataset, reconstructed with the SART algorithm. This reference dataset was used to compute both PSNR and SSIM metrics for our various scenarii. The same detector, and the same X-Ray generator as in section 5.4.2.1 were used, and the acquisition parameters were this time set to 120kV and 10mA.

We designed two distinct virtual acquisition scenarii for this experiment:

IAS - Increased Angular Step, which corresponds to a set of quasi equidistant projections covering a total angular range close to 360 degrees, resulting in a regular undersampling along the gantry rotation axis.

DAR - Decreased Angular Range, which corresponds to a contiguous set of projection, which were acquired with an angular step of 1°, where the whole set covers an angular range inferior to 360°, this strategy being also known as limited angle tomography.

Those two scenarii were derived for various number of projection taken from the original dataset, and we compared reconstruction results based on ssim and psnr for our DT$\mathbb{C}$WT based algorithm using a set of parameters derived from a previous set of experiments where we performed a regularization parameter sweep. The results of this experiments are presented on figure 5.10.

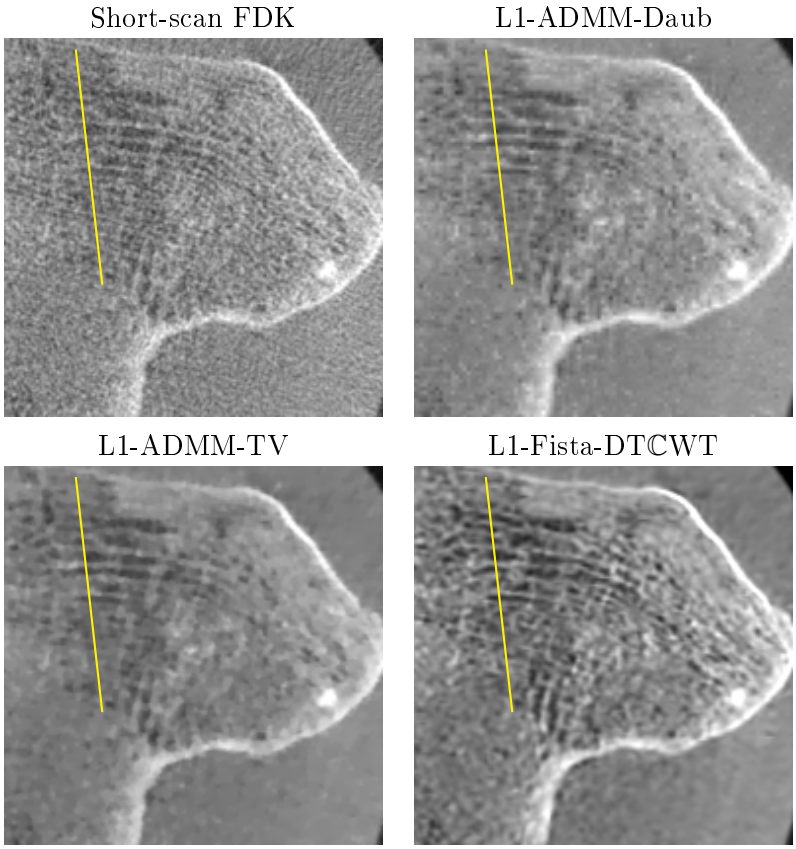


Figure 5.8: Visual overview of the reconstruction of a real human knee specimen using 4 different methods

5.4.2.3 Dose monitoring and acquisition strategy

The conscientious physicist, may argue that noise statistical properties may vary according to detector sensitivity, human tissues properties, and obviously X-Rays characteristics. Although it was physically impossible to challenge all these parameters, we designed a simple experiment, were we tested multiple X-Ray source features (kV and mA) while monitoring the total dose sent during the acquisition process, using a radiation dosimeter, kindly provided by Ronan Guillamet, from CEA.

Here the reference used to compute PSNR and SSIM score was acquired at the maximum dose, ie 100kV, 24mA. In all cases, we chose the DAR protocol, with a 190° total angular range, and the reconstruction algorithm was our DTℂWT, with the very same parameters obtained as stated in section 5.4.2.2. One can argue that different level of noise, may require different data fidely weighting term. Unfortunately, performing a full parameter sweep for each experiment was too time consuming.

The results of this experiment are presented on figure 5.11

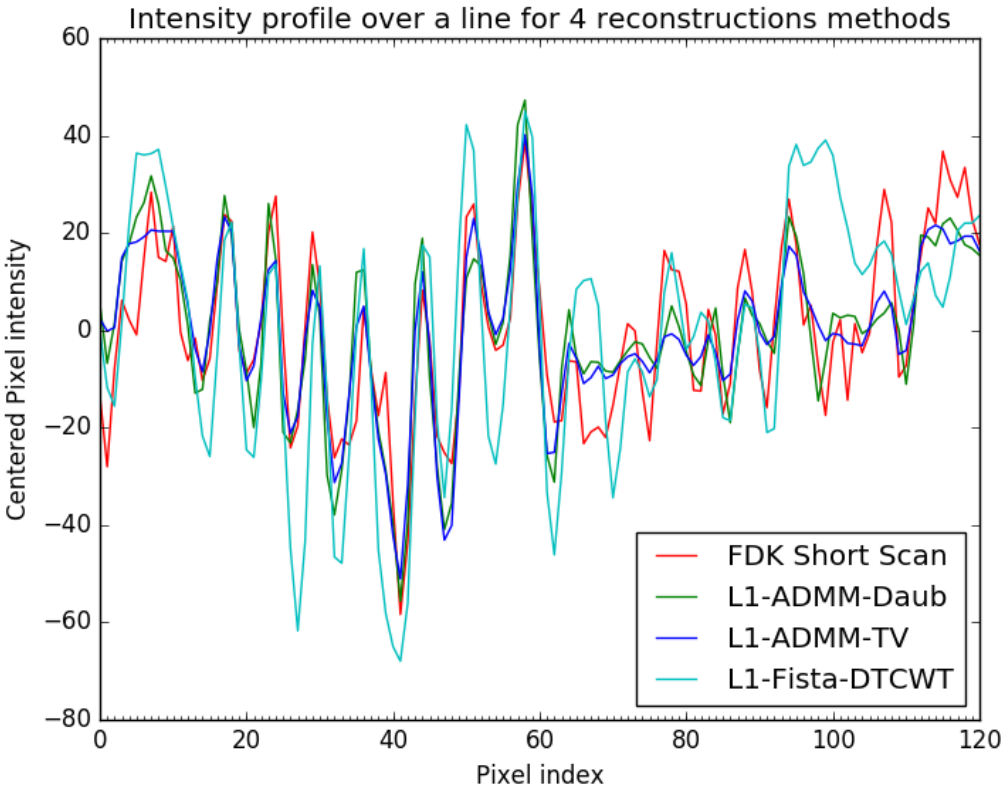


Figure 5.9: Yellow line profile from the 4 images presented in 5.8

5.4.3 DT$\mathbb{C}$WT Implementation performances

As stated in 5.3.4, we also studied the DT$\mathbb{C}$WT from a practical point of view, and derived a moderately optimized multiGPU implementation.

We reported the performances results we obtained, in terms of run time on various multi-GPU platform, challenging the workload size everytime. Our medium resolution dataset (512^3) containing a human knee CT medical image, yielded the results in 5.12. We also used a challenging dataset, with a high resolution volume of 1024^3 voxels, see 5.13.

5.5 Discussion

5.5.1 DT$\mathbb{C}$WT as a regularizing tool for CBCT reconstruction

The numerical experiments we performed on the synthetic dataset designed in 5.3.5 clearly showed the superiority of our proposed algorithm for the accurate retrieval of textured directional patterns in 3D. The fact that the total variation model was not the best signal sparsity

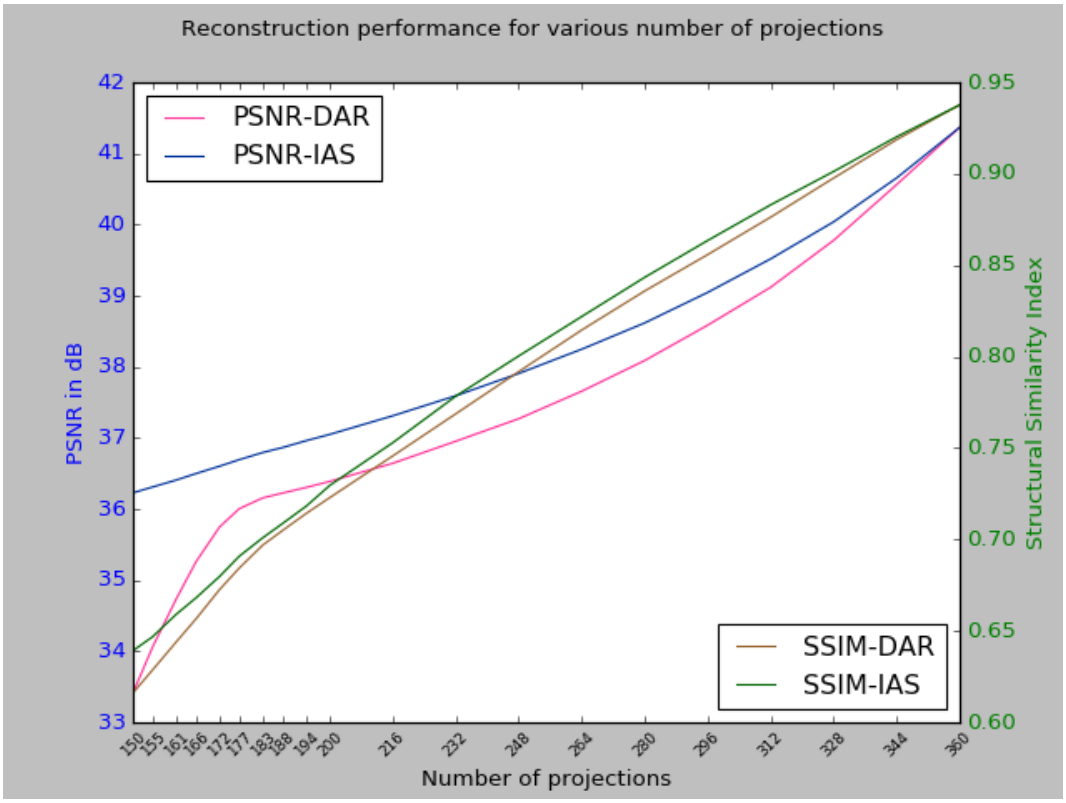


Figure 5.10: PSNR and SSIM for two acquisition scenarii (Decreasing angular range, and increasing angular step)

model for retrieving the Marschner-Lobb phantom is not surprising, given that our phantom is no more piecewise constant. As the signal to be recovered had bounded variation, we could have used a Sobolev pseudo-norm (l_2 instead of l_1 norm of the gradient module image) to regularize the problem, however, as our final target was the reconstruction of real CT-images, we chose the TV model as it is known to perform well in a wide range of imaging modalities [**pan2009commercial**]. The superiority of the DT$\mathbb{C}$WT model over the Daubechies wavelet model for inverse problem regularizatin in imaging, is also known for a long time, but its use in the framework of Cone Beam CT is rather new. Indeed, the CBCT images noise, and linear projection model is in practice different from the one used in the denoising / inpainting / deconvolution and its singular values distribution may differ greatly from those of these classical task, hence the possibility that it could have resulted in a correct but impractical optimisation method. Fortunately, we found that our approximately orthogonal wavelet tree based sparsifying operator performed quite well in practice, and allowed for a fast optimization algorithm implementation, capable of performing 100 iterations in about 18 minutes, for a 512^3 volume.

Another interesting aspect we discovered when performing our experiments was the particular behaviour of our algorithm when overestimating the sparsity of our image in the prior, see images in 5.6. Most of the sparse regularizer in real settings tend to oversmooth the resulting images, giving them a cartoon appearance, like the total variation model, or even adding aliasing artifacts for non redundant wavelet decomposition. When challenging noise

levels are being investigated, like the one used in our experiments, it can be seen that the TV and Daubechies model, does not seem to remove completely the "noisy" patterns in the image, although they do remove most of the high frequency signals in the image. To the contrary, it appeared that our algorithm was able to recover visually pleasing images, even in the case of sparsity overestimation: most of the noise is removed, and although part of the high frequency signal disappear, there does not seem to be unnatural pattern arising in the image.

This latter observation suggests that the coherence between the DT$\mathbb{C}$WT basis and the patterns arising from noise inconsitency in the data is low compared to the other invastigated models. The concept of uncertainty and mutual coherence, and their role in the framework of inverse problem in signal processing has been described in [**donoho2001uncertainty**]. The authors in this paper summarizes the role of mutual coherence in the uncertainty principle as follows: "If two basis are mutually incoherent, then no signal can have a highly sparse representation in both basis simultaneously." hence our earlier remark.

5.5.2 Real dataset experiments

Due to the lack of ground truth, we were not able to give PSNR and SSIM measurements for our real dataset. What we can say however from the results on figure 5.8, is that we are probably observing the same behaviour experienced in the previous synthetic data experiments regarding the ability of the algorithm to resolve small oriented structures while reducing the noisy pattern on the image.

One can see, by looking at the figure 5.9, that our algorithm was able to recover highly contrasted bone microstructures, even in the presence of linear system inconstistency due to modeling error, like beam hardening and noise in the data.

5.5.3 Undersampling strategy

Although theoretical results for optimal acquisition trajectory have been derived in the framework of analytical reconstruction, see for instance [**tuy1983inversion**], it is always interesting to take a look at empirical results obtained with iterative reconstruction technics provided on figure 5.10. In our case, it can be noticed that the PSNR of reconstructed images in case of limited angle, experiences a sharp increase between 150° and 180°, which can be explained easily using Tuy's conditions in our setup. PSNR being a logarithmic scale, one can notice that a linear increase in PSNR yield an exponential increase in $l2$ discrepancy, which is undeniably better than the square root increase in SNR predicted in the case of a Gaussian additive noise model. However the SSIM metrics, probably due to the fact that it is calculated as a mean, exhibit a more subtle difference for the DAR protocol between the $< 190°$ angular range and the $> 190°$ angular range: although the increase in SSIM with the number of view seems linear in both case, it appears that the slope in slightly higher when Tuy's conditions does not hold for the very short scan ($< 190°$) trajectories.

When using DAR protocol with more than a short scan trajectory, one can notice that the increase in number of views yields increasingly large return in PSNR, which clearly goes against the usual additive gaussian noise model. This unusual behaviour is still not well understood, and can arise from the fact that the actual noise model in the image is not dominated by gaussian noise, but can also be interpreted in terms of linear algebra. An increasing number of view can yield a linear problem with a decreasing condition number, and, as most first order method have an exponential convergence rate depending on the condition number, the fact that we experience such profile with a finite number of iteration seems more reasonnable.

The IAS protocol yielded curves that did not exhibited remarkable values, in both PSNR and SSIM, although the same remark we made for the DAR PSNR curve for the more than a short scan trajectories also holds.

When considering the practical interest of short scan trajectories for low cost equipments, we decided to use the DAR protocol with a total range of 190° in subsequent experiments.

5.5.4 Dose monitoring and acquisition strategy

When designing a practical acquisition protocol, one may be interested in assessing the influence of photon energy for a given X-Ray dose. This is exactly why we designed the experiment presented in section 5.4.2.3, whose results were reported on figure 5.11.

It must be noticed that comparing reconstruction from acquisition made at different photon wavelength is theoretically a nonsense, because it amounts to compare physically different attenuation properties. Unfortunately, we experienced this issue when comparing PSNR of images computed using a reference obtained at a different wavelength: PSNR of images acquired at 70kV were always approximately 5dB lower than PSNR of images acquired at 100 kV, like the reference image, independantly of the total dose.

However, material properties are generally not completely uncorrelated from one wavelength to another, so it can make sens to compare visual informations from images acquired at different wavelength, which is what we tried to do by computing the SSIM.

Our experiment showed that increasing dose linearly at 70kV and 100kV yielded decreasing returns in terms of SSIM, allowing to choose a reasonnable trade-off between 100kV-7mA and 70kV-25mA. However, for a given dose in mGray, increasing the X-Ray voltage to 120kV yielded images differing too much from our reference image (around 0.2 in SSIM), whose low quality was validated by visual inspection.

5.5.5 Implementing DT$\mathbb{C}$WT on multiple GPU

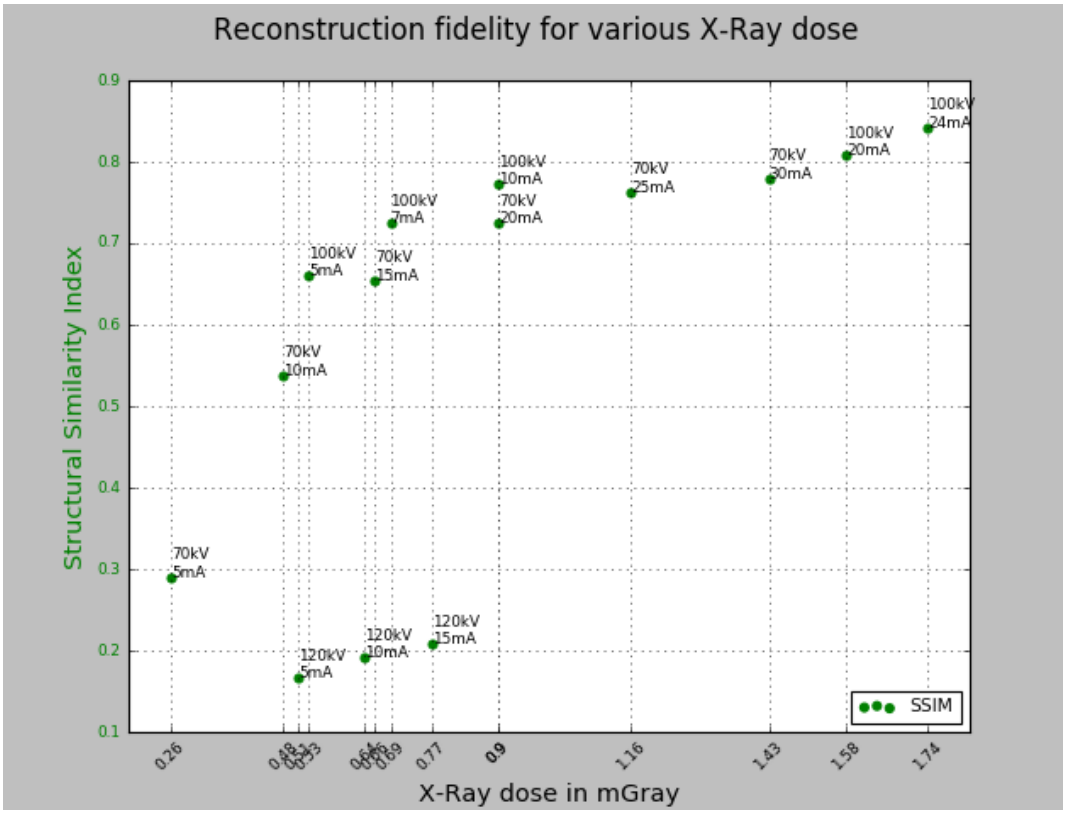
The results presented on figure 5.12 and 5.13 suggest that a relatively fast implementation of the DT$\mathbb{C}$WT transform can be obtained using GPUs. Our fastest setup was able to perform sequentially analysis and synthesis operation in less than 6 seconds for

the 512^3 dataset. Comparatively, peforming the same operation, with the same data, and same filters using the python/numpy implementation provided by the opensource package [**DTCWTOpenSourceImplem**] on a Intel core i7 3970X, with 64GB Ram took about 200 seconds.

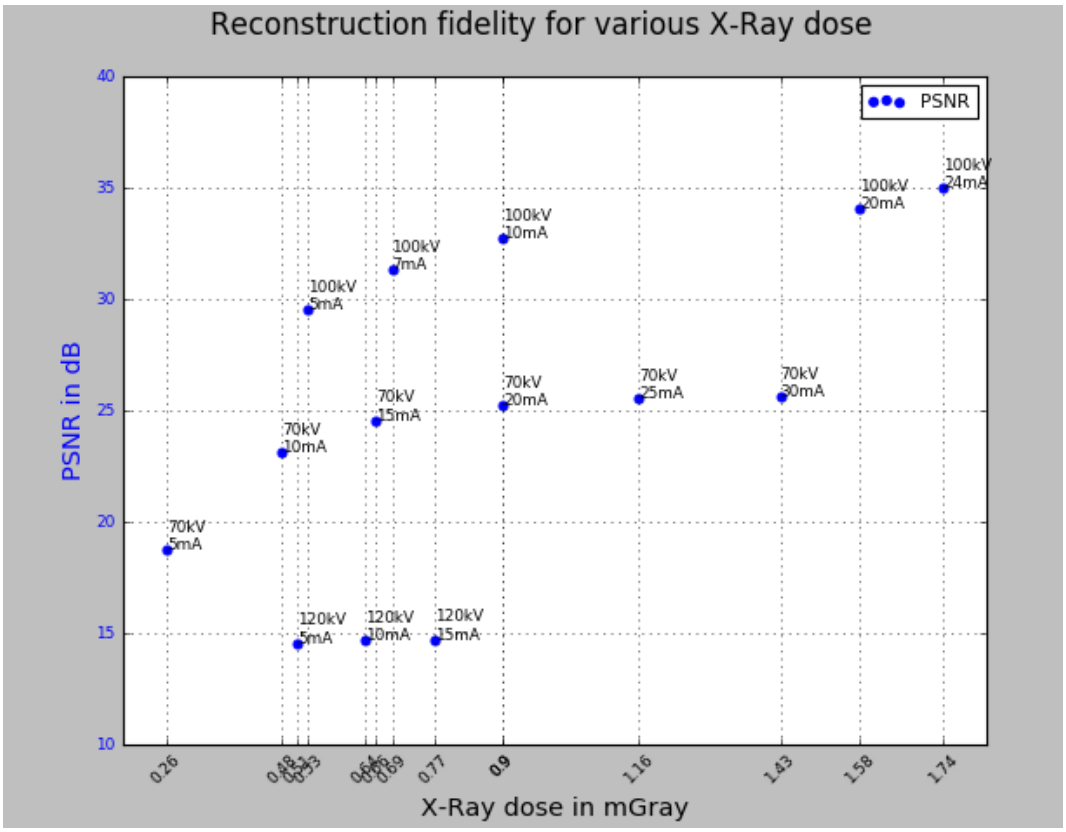
Although the 33 times speedup regarding a possibly non-optimized CPU implementation was an interesting result, the scalability of our solution along with the number of GPU was rather poor. The speedup obtained while getting from one GTX970 to two GTX970 for instance, was only 12.5 % on the fastest instances for both datasets. Investigating this issue revealed that a lot of time was spent in copying data between host and devices, the ratio between compute time and host-device copy time prevented us from parallelizing computations efficiently.

5.6 Conclusion and Future work

The results of the present study shows practical feasibility of sparse regularization of CBCT reconstruction using a directional and separable complex wavelet transform. In our specific synthetic use case, exploiting sparsity prior in the DT$\mathbb{C}$WT domain outperformed significantly ($>$ 1dB in PNSR) two other algorithm based on total variation and Daubechies wavelets sparsity models. In the real dataset experiments, the DT$\mathbb{C}$WT yielded visually better results than the other methods, especially in the task of reconstructing human knee bone microstructures. Our study showed that the DT$\mathbb{C}$WT was also well suited for a GPU implementation, our moderately optimised implementation of the transform allowed for a total reconstruction runtime of about 18 minutes for 100 iterations over a 512^3 volume. Although we did not experienced convergence problem in both synthetic and real dataset, we will probably use a more relevant optimisation framework in the future, in order to properly address the quasi tight frame featured by our DT$\mathbb{C}$W tree, using work developed in [**pustelnik2012relaxing**], or other framework for analysis formulation, as in [**chai2007deconvolution**], [**li2008iterative**], [**cai2009linearized**] and [**cai2010framelet**]. Another aspect that clearly needs to be studied in the framework of 3D CT imaging, is the role of structured sparsity models, like those presented in [**rao2011convex**], in the recovery of human body structures under noisy of incomplete measurements. We will probably try to address this issue in a future work.



(a) Mean structural similarity index for various dose



(b) Peak signal to noise ratio for various dose

Figure 5.11: PSNR and SSIM for various acquisition scenarii (X-Ray generator settings)

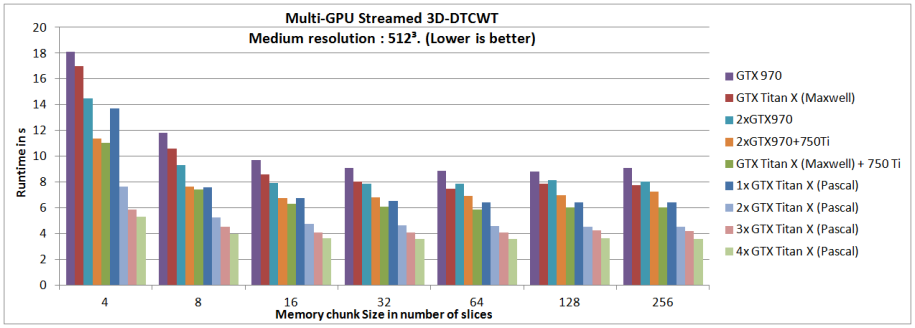


Figure 5.12: Runtime for the 3D DTℂWT transform, on a 512^3 dataset

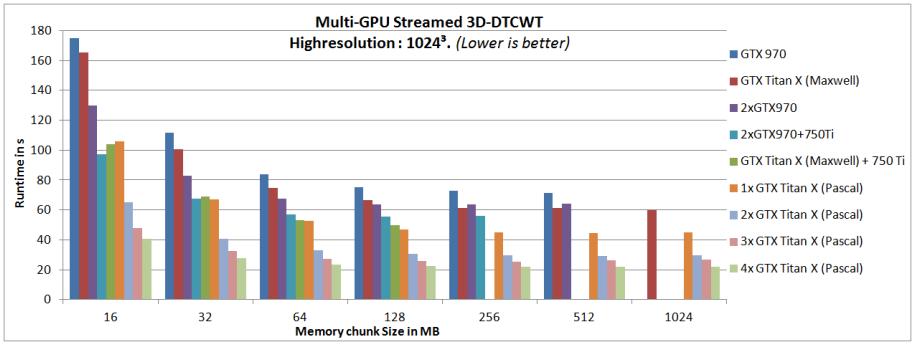


Figure 5.13: Runtime for the 3D DTℂWT transform, on a 1024^3 dataset

Conclusion

Computerized tomography is a very broad topic of research, ranging from physics, with the study of X-Ray interactions with matter, to duality theory and monotone operators framework for convex optimization. CT has benefitted from advances from many of the fields of science and engineering in the past few years. Regarding reconstruction algorithm in particular, most of the recent advances were made possible thanks to the advent of algebraic methods, providing a versatile framework to experiment high fidelity tomographic projection models, noise models, allowing to take into account multiple energy, photon scattering, ...

In this thesis, we tried to explore a few discretization models based on regular grids, and various smooth and non-smooth volume elements. Although the theoretical properties offered by non standard grid like the Body Centered Cubic grid were appealing, the fact that work in this field generally remains almost unnoticed discouraged us to investigate further. Indeed, there does not seem to be any available software able to handle 3D image data on BCC grids, such that comparing images reconstructed with CC grid and BCC grid is almost impossible without introducing a methodology bias related to grid to grid interpolation process.

It must also be noticed that defining interpolation methods with a reasonable quality, ie for instance spline of order >1 is nontrivial, and requires implementing a BCC grid based discrete Fourier transform and its inverse.

Another argument that discouraged us to investigate further on non standard grids and volume elements is that, they are generally not compliant with advanced frame based analysis operations. For instance, one must notice that their are currently no dilation matrices that allow to build a proper multiscale decomposition in the same fashion wavelets has been built on cartesian grids.

A second aspect explored in this thesis was related to first order methods for tomography. It is interesting to notice that the batch based approach derived in the early 80's fort OSEM and SART in tomography met a great success in the field of large scale optimization. Of course, recent advances are mostly targeting differentiable objective like the least square, and are based on theoretical analysis of expected cost in the framework of stochastic optimization. However, it is not clear if stochastic optimization may benefit to CT reconstruction, because input datasets, at least in CBCT are not expected to grow at the same rate as the learning datasets in machine learning in the years to come.

Regarding the choice of tomographic operators, there seems to be different approaches wether one is targeting a good methodology, or a fast method for routine experiments. The conscientious scientist that wants to monitor convergence speed of a given algorithm, or the primal dual objective gap, in order to deliver optimality certificate will probably be interested in using proper adjoint operators for the backward projection when instanciating optimization algorithms. Otherwise, one can use some tricks in order to obtain the fastest algorithm, like fast but aliasing prone back projector, and FBP or FDK algorithm in place of backprojection,

... These tricks are not always backed by a strong theoretical background, but may result in valuable accelerations in practice.

Finally, regarding the use of sparse priors in regularized iterative tomography, we proposed a simple but efficient algorithm in our last chapter, that appeared to outperform two regular methods from a well known open source software package. The fact that redudant frames, with shift invariant properties provides a good model for sparse regression is not new, however setting up such methods for high dimensional dataset in a reasonnable amount of time was not a trivial task. We provided a multi-GPU implementation of the 3D dual-tree complex wavelet transform, able to run on heterogeneous set of GPUs, although this implementation suffered from severe host to GPU copy overhead, and could benefit from a proper optimization process.

In the near future, we postulate that analysis formulation of sparse regression problem will benefit from more complex sparsity models, eventually mixing non local total variation, dictionaries, redundant and anisotropic wavelets trees for instance. Extension of supervised methods from dictionary learning to adaptive filtering may also be an interesting lead, for instance, recently, nonconvex objectives, based on filters-like neural networks were designed in order to regularize tomographic reconstruction, see [**kang2016deep**].

One recent advance in tomography that attracted our attention during this thesis is the Differential Phase Contrast Cone-beam CT (DPC-CBCT, see [**fu20153d**]), although this method seems promising, it appeared that the phase grating shifting process rely on precise mechanical movement (less than a few μm) that precludes this method to be available on low cost equipments in a near future.

However, recent advances in on-line CBCT system calibration, based on epipolar consistency conditions seems promising, and easy to implement in low cost CBCT systems, which may allow for a wider availability of CBCT systems, equiped with analytical and algebraic reconstruction software, such as the one designed in the framework of this thesis.

Résumé La tomographie est une technique permettant de reconstruire une carte des propriétés physiques de l'intérieur d'un objet, à partir d'un ensemble de mesures extérieures. Bien que la tomogaphie soit une technologie mature, la plupart des algorithmes utilisés dans les produits commerciaux sont basés sur des méthodes analytiques telles que la rétroprojection filtrée. L'idée principale de cette thèse est d'exploiter les dernières avancées dans le domaine de l'informatique et des mathématiques appliqués en vue d'étudier, concevoir et implémenter de nouveaux algorithmes dédiés à la reconstruction 3D en géométrie conique. Nos travaux ciblent des scenarii d'intérêt clinique tels que les acquisitions faible dose ou faible nombre de vues provenants de détecteurs plats. Nous avons étudié différents modèles d'opérateurs tomographiques, leurs implémentations sur serveur multi-GPU, et avons proposé l'utilisation d'une transformée en ondelettes complexes 3D pour régulariser le problème inverse.

Mots clés : Tomographie, GPGPU, problème inverse, parcimonie.

Abstract

X-Ray computed tomography (CT) is a technique that aims to provide a measure of a given property of a physical object interior, given a set of exterior projection measurement. Although CT is a mature technology, most of the algorithm used for image reconstruction in commercial applications are based on analytical methods such as the filtered back-projection. The main idea of this thesis is to exploit the latest advances in the field of applied mathematics and computer sciences in order to study, design and implement algorithms dedicated to 3D cone beam reconstruction from X-Ray flat panel detectors targeting clinically relevant usecases, including low doses and few view acquisitions. In this work, we studied various strategies to model the tomographic operators, and how they can be implemented on a multi-GPU platform. Then we proposed to use the 3D complex wavelet transform in order to regularize the reconstruction problem.

Keywords: Tomography, GPGPU, Inverse problem, Parcimony

GIPSA-Lab, 11 Rue des Mathématiques
Saint-Martin-d'Hères